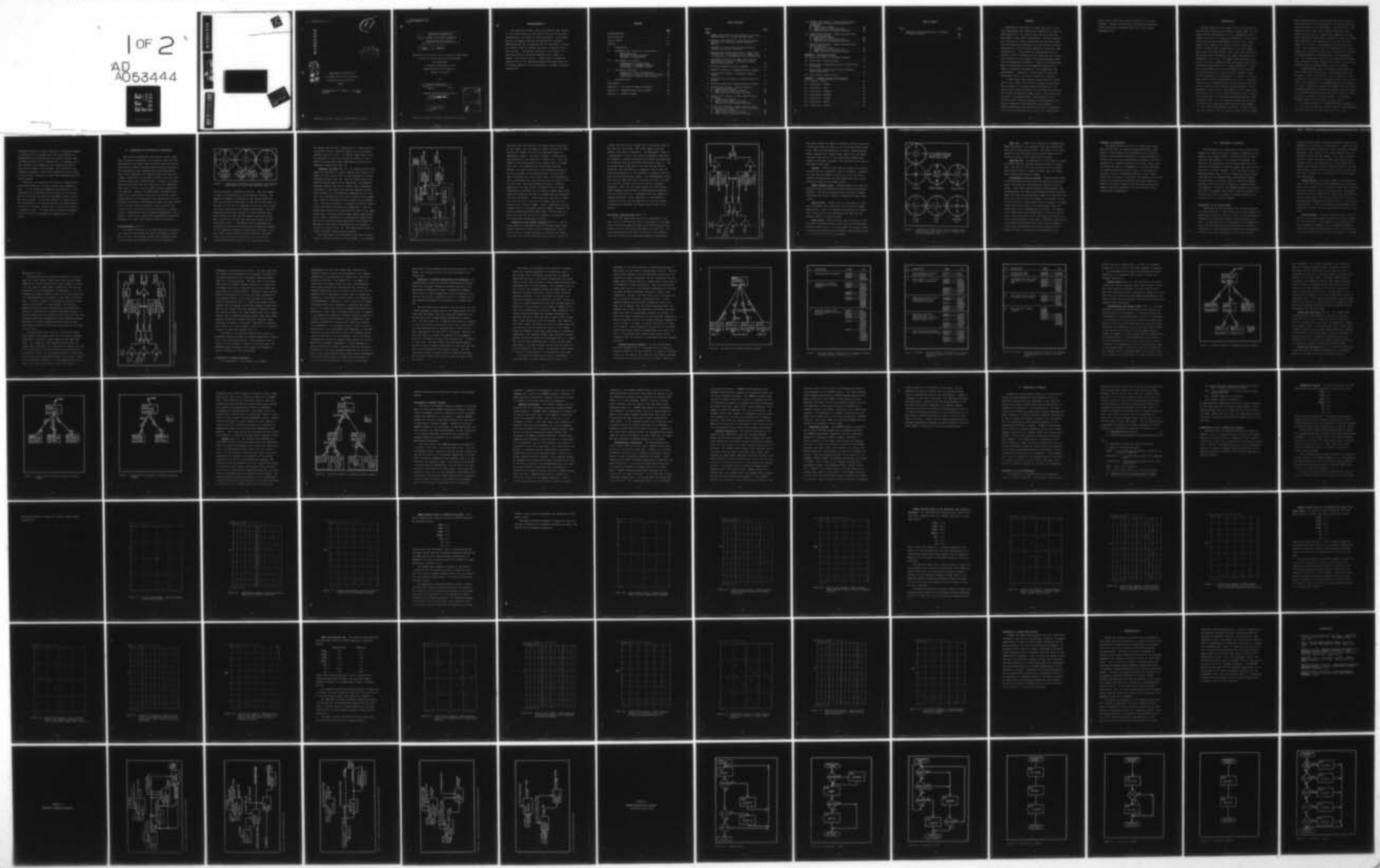
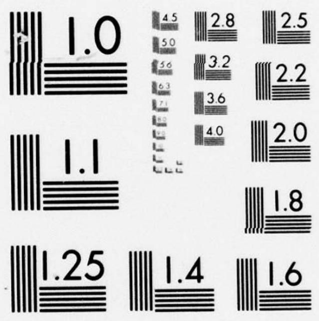
AD-A053 444 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 17/2
QUANTITATIVE PROCESSING OF SIGNAL SPACE DATA TO PROVIDE TRANSMI--ETC(U)
DEC 77 R A MINTONYE
UNCLASSIFIED AFIT/GE/EE/77-28 NL

1 OF 2
AD
A053444




1.0
4.5
5.0
5.6
6.3
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

AFIT/GE/EE/77-28

(1)

QUANTITATIVE PROCESSING OF SIGNAL SPACE DATA TO PROVIDE TRANSMISSION LINE ANALYSIS

THESIS

AFIT/GE/EE/77-28 Robert A. Mintonye
Captain USAF

$12 . 25

14 AFIT/GE/EE/77-28

6

QUANTITATIVE PROCESSING OF SIGNAL SPACE DATA TO PROVIDE TRANSMISSION LINE ANALYSIS.

9 Master's THESIS,

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

10 Robert A. Mintonye, B.S.
Captain USAF

Graduate Engineering Electronics

11 Dec~~ember 1~~977

12 135 p.

012 225

bb

Acknowledgements

The engineers at RADC, Codex, and IBM were very helpful in the initial efforts on this thesis and much of the basic system design is a result of their assistance. The development of the program structure design and the resultant FORTRAN program was accomplished with the assistance of Capt. K.L. Marvin, Capt. R. DeSanto, and Capt. R. Herron, fellow AFIT students. The general direction of the thesis development and technical writing was guided by Dr. (Capt.) G. Vaughn, AFIT faculty advisor. I would like to express my appreciation for the assistance provided by these people and to my wife, Maureen, for her assistance in typing, proofing, and patience.

Contents

List of Figures

List of Tables

Abstract

Quantitative processing of signal space data to provide transmission line analysis was developed in support of work being done at the United States Air Force Rome Air Development Center (RADC) laboratories. The RADC engineers were working with digital data modems and they were qualitatively analyzing transmission line perturbations through observation of an oscilloscope display of the signal space representation. The display was being generated by an optional circuit available with the modem, but the oscilloscope display was not capable of providing adequate quantitative perturbation information. However, the signal space data did contain all of the information required for a complete quantitative analysis of the transmission line perturbations. Therefore, a system using the same data potentially could be devised to perform the analysis.

A signal space data analysis system design was developed in this thesis, which would be capable of providing three displays of the signal space representation, the amplitude and phase deviations, and the frequency spectrum of the phase deviations or phase jitter. The system would consist of the digital modem, a minicomputer, and an interface device between the modem and minicomputer which would be capable of interrupting the minicomputer. A FORTRAN program was also developed and run in a simulation effort which transformed the xy data of the signal space into a form of data which could be plotted on the three displays. Though the system was not actually constructed, it was successfully simulated such that it would merit implementation.

I. Introduction

Through analysis of the changes in the coordinates of a received data communications signal in signal space, one can obtain several characteristics of the transmission line or channel over which the signal was transmitted. This is possible because channel perturbations such as noise, phase jitter, hits, harmonic distortion, and line outages will affect the signal space representation of a data signal in unique and consistant manners. Also, since most systems use data signal point randomizers, the signal space pattern is independent of the data. The signal space patterns of an active line can be analyzed without knowing the data content and without interfering with the data transmission. Thus, the line perturbations can be determined without removing the line from service as must be done to perform conventional transmission line measurements. However, the signal space analysis must be performed in real time to completely obtain the line characteristics, and the required speed has not been practical until the low cost development of minicomputers. Practical real time analysis of the signal space for data rates up to 9600 bits per second has been possible since 1976 within the International Business Machines (IBM) corporation (Ref. 1), but a method of performing the analysis has not been made available outside the IBM company.

This thesis was developed to provide a system of analysis which could be used to analyze in real time the signal space representation of a data signal and which could provide quantitative line characteristic outputs much like the real time outputs of the IBM system. The signal space coordinates could be taken from a digital modem receiver in binary form and fed to a computer for comparison with the known transmitted signal space coordinates (since the signal space coordinates are independent of the data, the data symbols do not need to be determined to perform the analysis). The difference between the transmitted and received signal space coordinates could then be analyzed to determine the characteristics of perturbations on the transmission line. The signal space data could be taken from the modem via an external transfer register which could be read by the computer when an interrupt flag is set by the modem.

A FORTRAN program which is capable of analyzing the signal space data and providing a quantitative output was developed for this thesis but it must be expanded to work optimally on the particular minicomputer which will be utilized in the laboratory. The FORTRAN program was developed for use on a multiuser computer system which did not have an interrupt capability, therefore the program does not contain the interrupt handling routines required to read the input data and to control the output displays. The required program expansions are discussed in Section III.

The output created by the FORTRAN program consists of real time printer displays of signal space, of the amplitude and phase deviation for each data point, and of the frequency spectrum for phase deviations. Though the program is currently able to provide only real time displays of transmission line perturbations, the analysis system would be greatly enhanced if the capabilities of averaging, creating, storing, and displaying a time condensed history of line data are added to the program. The structure of the program is designed so that these capabilities can be added to the existing instructions rather than reaccomplishing the entire program.

The thesis is organized with sections on problem background and definition, solution, results and recommendations. In the section on background and definition of the requirement is a description of the origin of the requirement and the practical aspects of equipment which affected the system design. In the next section is a description of the development of the system structure and program design, followed by a separate section which centers on discussion of program results. The final section contains recommendations for further development and implementation of the system.

II. Background and Definition of Requirement

The need for quantivative processing of signal space data to provide transmission line analysis arose from work being done at the United States Air Force Rome Air Development Center's (RADC) telecommunications digital laboratory. The telecommunications engineers were doing development work in the laboratory with digital modems which were providing a qualitative oscilloscope picture of the real time, received signal space representation. Ideally, the signal space picture could show the condition of the transmission line, and it could be used to detect several transmission line problems. However, the qualitative information provided by the oscilloscope could only be utilized to give a rough indication of transmission line quality since it only provided an average of data which the human eye could detect. Even a trained person watching the scope could not tell whether or not the line was about to experience a lineout (loss of transmission) due to a gradual degradation in line quality. Thus it was decided that a quantitative method of analyzing the signal space pattern should be developed.

Modem Parameters (Ref. 2)

The modems being used in the RADC laboratory are primarily Codex LSI 9600 modems and are on consignment to RADC. They are full duplex digital modems with quadrature phase shift and amplitude modulation at a center frequency of

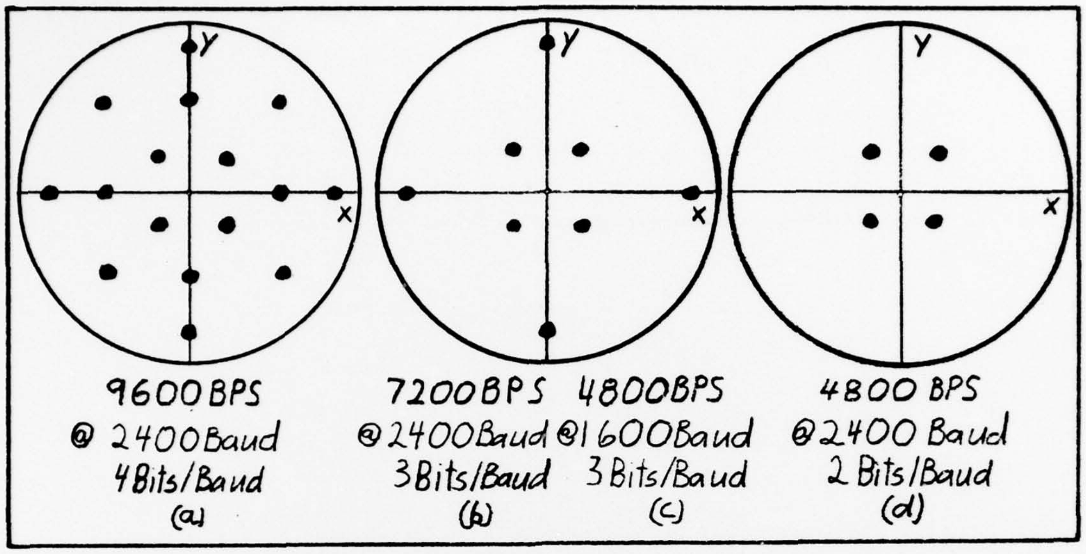


Figure 1. Signal Space Patterns (Eye Patterns) for Different Data Modes on a Codex 9600 LSI Modem (Ref. 2)

1706 Hz and with a bandwidth of 2400 Hz. They are designed to work on C2 Conditioned 4 Wire telephone circuits. The modems are capable of operating at data rates of 9600 BPS (bits per second) at 2400 baud, or 7200 BPS at 2400 baud, or 4800 BPS at 1600 baud, or 4800 BPS at 2400 baud all of which are switch selectable on the modem control panel. The corresponding signal space pattern for these different data rates contain 16 points for the highest data rate of 9600 BPS, 8 points for the next two rates and 4 points for the lowest rate (Fig. 1). The modems contain equalizers for adjusting the delay and for equalizing the amplitude between the two quadrature components during each baud time. They also contain data randomizers which rotate the different signal space points for each baud so that the data is uniformly distributed among all the signal space points.

The signal space pattern is generated by a Codex system as an optional circuit inside the modems or as an optional external plug-in circuit. The digital signal which feeds the generator is taken from inside the modem as x and y values of signal space after demodulation and delay and quadrature equalization (Fig. 2).

<u>Operation</u> <u>of</u> <u>Modem</u> (Ref. 2). The analog received data signal is converted to a serial digital data string in the sample and hold circuit prior to the demodulation section (digital multiply) (Fig. 2). Then the serial data is demodulated using digital filter algorithms. The adaptive equalizer adjusts each point of signal space information (equivalent to each baud) for delay and amplitude equalization between the two quadrature components as it is fed into the xy accumulator (the x and y values for the signal space point are multiplexed into a single word of information). Word synchronization is supplied by another circuit within the modem and signals the accumulator to feed the signal space word into the data decision logic at the appropriate time. The data decision logic simply demultiplexes the xy values, determines the closest preset signal space point to the received signal space point, and assigns the appropriate data value as determined by the randomizer circuit (not shown in Fig. 2). The data decision logic is simply a minimum distance decision system.

The signal space point coordinates are stored in ROM inside the modem and are set at the factory. As different

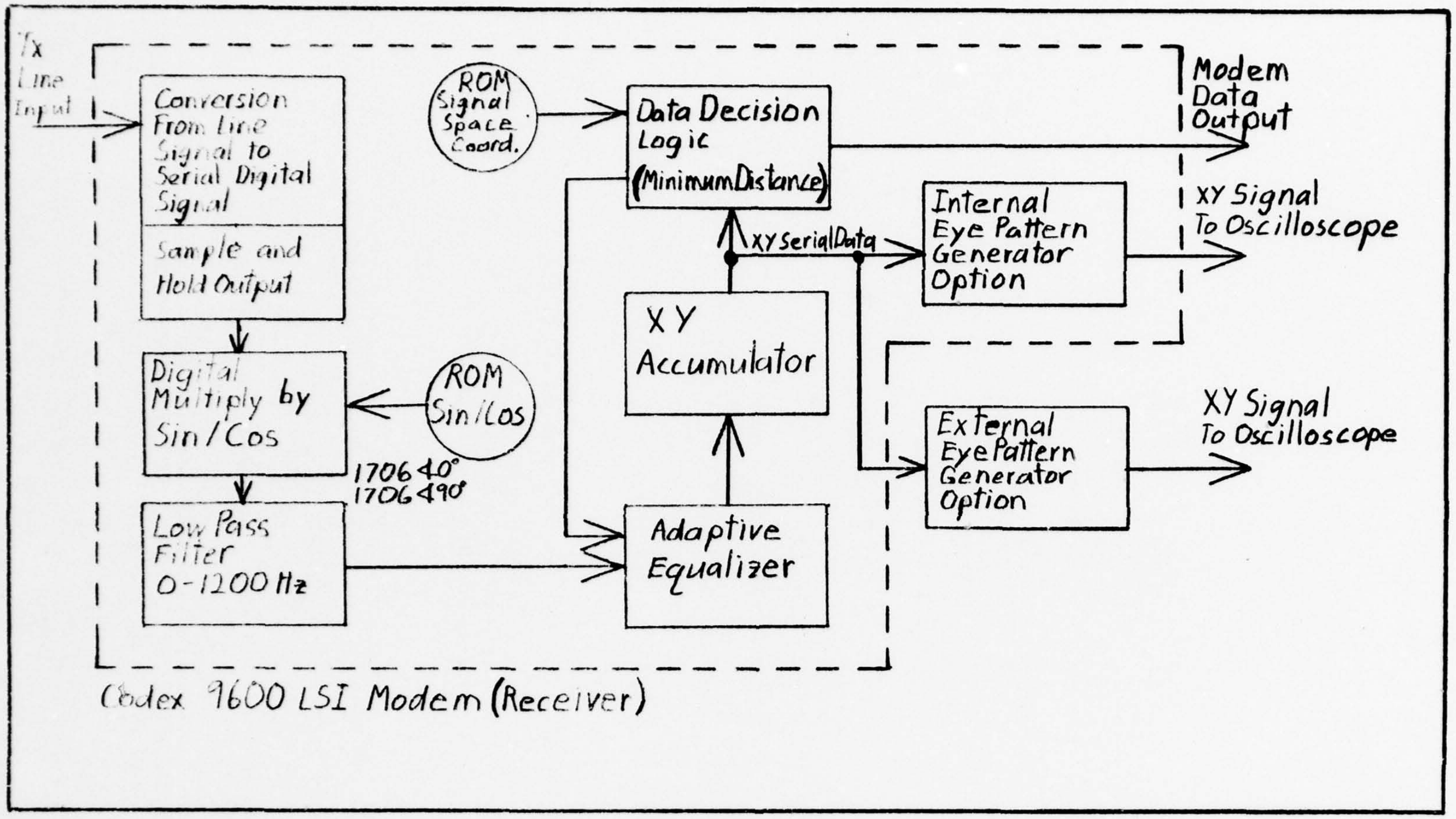

Tx Line Input
Conversion From Line Signal to Serial Digital Signal
Sample and Hold Output
Digital Multiply by Sin/Cos
Low Pass Filter 0-1200 Hz
ROM Signal Space Coord.
ROM Sin/Cos
1706 40°
1706 490°
Data Decision Logic (Minimum Distance)
XY Accumulator
Adaptive Equalizer
XY Serial Data
Internal Eye Pattern Generator Option
External Eye Pattern Generator Option
Modem Data Output
XY Signal To Oscilloscope
XY Signal To Oscilloscope
Codex 9600 LSI Modem (Receiver)

Figure 2. Partial Block Diagram of Codex 9600 LSI Modem Receiver with Codex Eye Pattern Generator Options

data rate modes are selected, the signal space coordinates do not change, only the number of points utilized for transmission are changed, and only the randomizer makes the appropriate changes. As shown in Figure 1 certain points are eliminated as the data rate is slowed down. Also, the number of bits represented by each signal point is changed from 4 to 3 to 2 depending on the data mode selected. However, this again only affects the randomizer mechanism and not the signal space mechanism since the same signal space coordinates are used in each mode. For transmission, the xy coordinates of a point are put into two words of five bits per word. Then the randomizer selects a different signal point for each baud of information and the modem transforms and transmits the two words of signal space coordinates as quadrature sin and cos frequencies with magnitude and phase information. At the receiver, two words of eight bits each are used to describe the signal space xy coordinates of the received signal. These coordinates are then compared with the coordinates stored in ROM to determine the closest transmitted signal space point. The two words of eight bits each are also used to generate the signals for the oscilloscope signal space picture.

<u>Operation</u> <u>of</u> <u>Eye</u> <u>Pattern</u> <u>Generator</u> (Ref. 2). The two Codex options for generating the oscilloscope signal space picture both use the same circuit configuration. The internal option is merely included on a plug-in card inside each modem case and the external option puts the circuit in

a small case that can be plugged into the external jacks on any Codex LSI 9600 modem. The Codex company calls the options "Eye Pattern Generators". The generators consist of a demultiplexer circuit, two serial to parallel converters, two digital to analog converters, and buffers and amplifiers for interface with the modem and oscilloscope (Fig. 3). The xy serial signal space data is fed into the Eye Pattern Generators from the modem xy accumulator (Fig. 2) and the word synchronization for the generator is supplied from the modem word synchronization signal. The demultiplexer breaks the serial xy data word into two words for x and y of eight bits each and strings them into the serial to parallel registers. Then, the word synch signal from the demultiplexer signals the converters to simultaneously dump the 8 bits of x and 8 bits of y coordinates into the digital to analog converters. These converters in turn generate a voltage level which can be amplified to drive an ordinary oscilloscope to generate the signal space pattern or eye pattern.

Eye Pattern Characteristics (Ref. 2)

Since the signal space points are independent of the data, the line perturbations will be contained in the signal fed to the oscilloscope and will not be affected by nor interfere with the data being transmitted over the line. The only perturbations which will not show up on the oscilloscope are differences in quadrature amplitude distortions

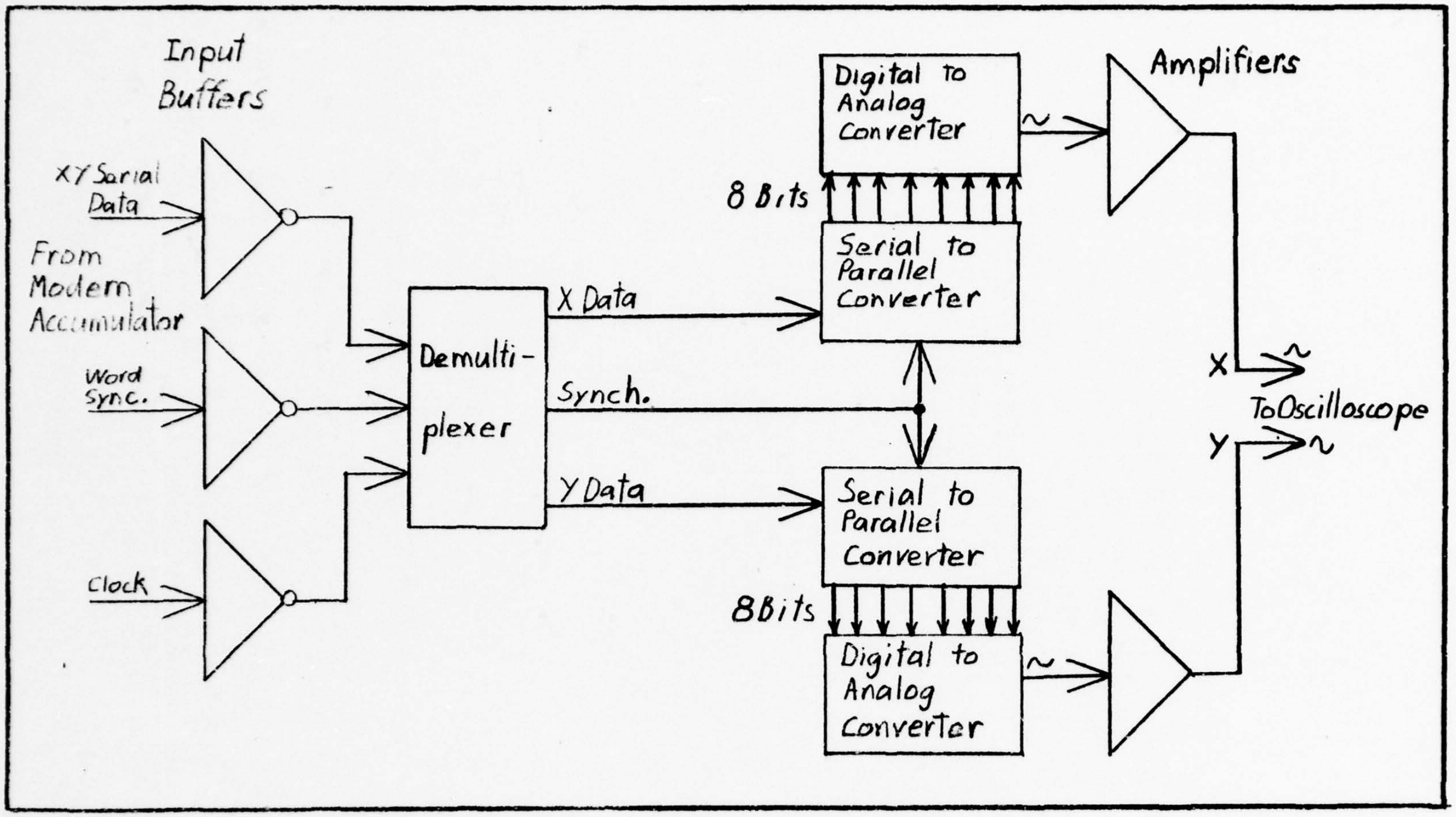


Figure 3. Schematic of Codex LSI 9600 Modem Optional External Eye Pattern Generator (Ref. 2)

and delay because the adaptive equalizer corrects them prior to the point where the data is fed to the Eye Pattern Generator. Pure line perturbations of lineout, normal gaussian noise, impulse noise, phase jitter, phase hits, and amplitude hits will show up on the oscilloscope signal space pattern as depicted in Figure 4 for 7200 BPS at 2400 baud or 4800 BPS at 1600 baud.

Lineout. Lineout occurs when the signal loses synchronization and the transmitter ceases to transmit. Therefore, the only received signal is the noise on the line which contains various phases and small amplitudes which center around the origin of the signal space.

Normal Gaussian Noise. Normal gaussian noise adds and subtracts from the transmitted signal amplitude and phase. Therefore, the received signal points will have a random scattering about their respective transmitted signal space points.

Impulse Noise. Impulse noise is very narrow in time and consequently only affects one or two received signal space points. Thus, the impulse noise appears as a momentary single spot on the oscilloscope far away from the general cluster of received points.

Phase Jitter. Phase jitter is the oscillation of the sequential transmitted phase points about their respective reference phase points. The oscillation will simply cause a rotational rocking motion of the entire signal space pattern as it is observed on the oscilloscope.

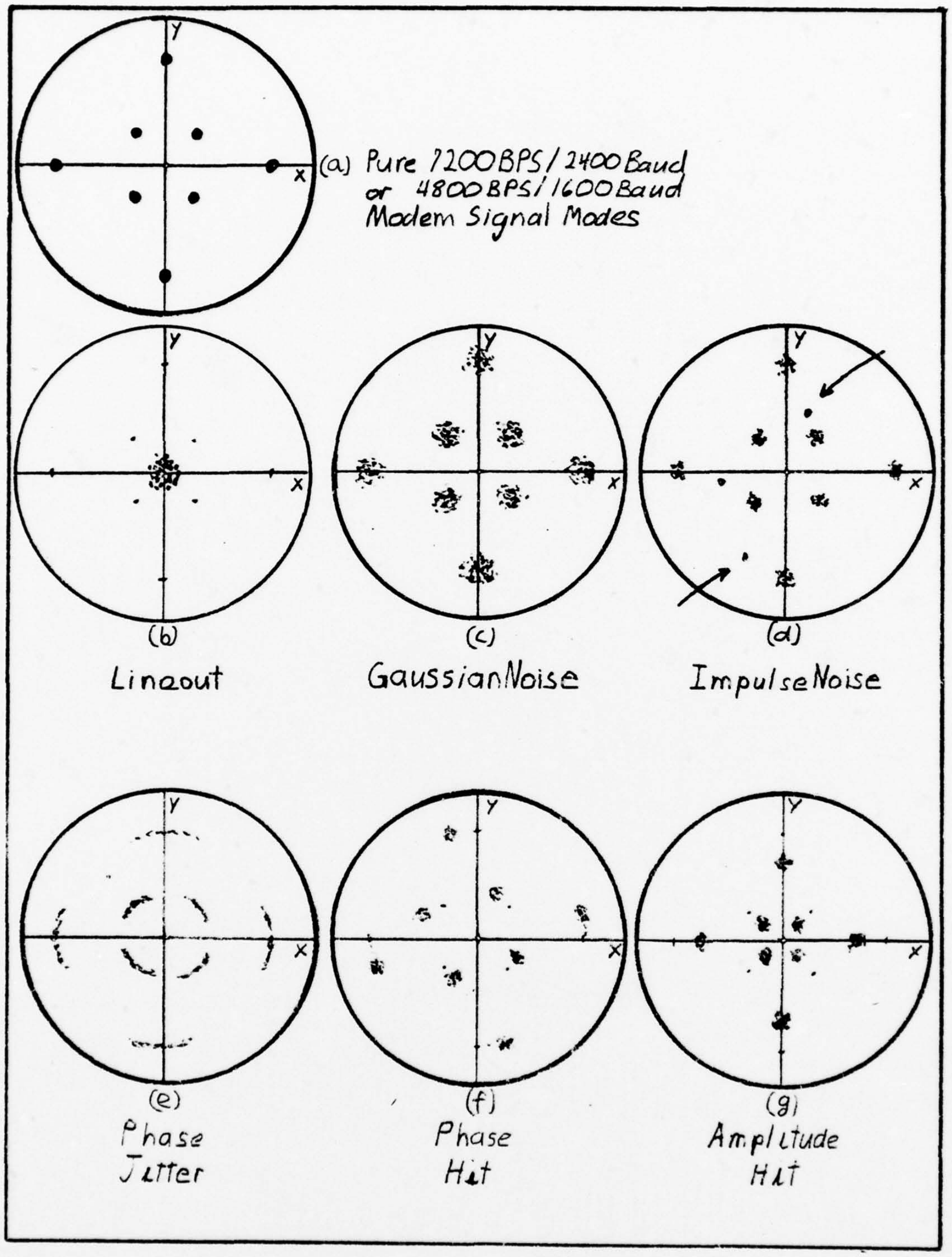


Figure 4. Oscilloscope Digital Eye Pattern (Signal Space Pattern) with Pure Signal and with Several Pure Line Perturbations (Ref. 2)

Phase Hit. A phase hit is caused by a constant phase angle being added to or subtracted from each transmitted point for a short period of time. The affect on the oscilloscope is that the signal space pattern rotates by the amount of the hit, then rotates back to normal.

Amplitude Hit. An amplitude hit is the same as a phase hit except that the constant deviation affects the amplitude. The affect on the oscilloscope is that the signal space pattern seems to expand or shrink about the signal space origin then return to normal.

Limitations of the Eye Pattern. Though the pure cases of perturbations are easy to see and detect on the oscilloscope, they cannot be quantitatively analyzed by an observer. Also, when several different perturbations occur simultaneously, the types of perturbations cannot be determined, and when the perturbations happen at a high rate, the human eye cannot follow the changes. For instance, if the line is noisy and a small phase jitter is present, the phase jitter will not be seen on the oscilloscope because the phase jitter will be covered up by the noise. If impulse noise strikes only very sporadically, it will not even be seen on the scope since one or two points will appear for only 0.4 milliseconds at 2400 baud. Therefore, the oscilloscope signal space picture can only be used to roughly detect line perturbations and it only indicates that a general problem is occuring on the transmission line.

Statement of Requirement

The primary problem is that the oscilloscope signal space picture does not provide enough quantitative information to fully analyze the transmission line condition. However, the information provided to the oscilloscope does contain most of the line perturbation information which is required for a complete analysis of the line. Also, since the modems are only on consignment to the RADC laboratory, no internal modifications may be made to the modems and therefore only the information available at the output points may be utilized. Therefore, the problem is to devise a method of quantitative analysis which uses either the digital eye pattern signal or else the analog eye pattern signal to provide quantitative information about transmission line perturbations.

III. Development of Solution

The development of a system which could quantitatively analyze the signal space data to obtain a transmission line analysis was straight forward. Interviews with the RADC, Codex and IBM engineers, and research of literature led directly to a formulation of the system design and to the decision of analyzing the digital signal space data from the modem with a digital computer. Since the basic concept of analyzing signal space data for transmission line analysis could apply to any digital modem using minimum distance decision logic, it was also decided to keep the program flexible so that it could be adapted to different modems and computers. Therefore, the FORTRAN program was developed such that it contained functionally structured elements of routines that could be adapted or expanded without major modification to the entire program.

Formulation of the System Design

During the initial interviews with the RADC engineers and Codex engineers, it was learned that the IBM company had published an article in late 1976 (Ref. 1) which described a transmission line monitor concept utilizing signal space data. An interview with the IBM engineers was arranged and their LQM (Transmission Line Quality Monitor) system was demonstrated. Their LQM system demonstration was impressive in that it provided a great deal of real time analysis outputs and of condensed historical numerical outputs on the transmission line quality and performance. However the LQM system was extremely complex and it was strictly an internal IBM system, thus details of the system operation were not available outside the IBM company. But, since a workable system of digital analysis of signal space data for line analysis had not been demonstrated, it was decided that a simpler system using the same concepts to provide only real time analysis outputs was within the scope of a thesis and that such a syatem could be developed for the RADC laboratory requirements.

System Concepts. The basic concept of the system design is that since most of the transmission line characteristics are contained in the two x and y words which describe each received data point in the signal space, the analysis system need only be comprised of a device for extracting the digital signal space data from the modem and a device for analyzing and displaying the processed data to show the line characteristics. This concept meets the constraints of not modifying the modems and of using raw data available at the modem.

System Hardware. The data extraction device would be similar to the Codex optional external Eye Pattern Generator, and the analysis and display device could consist of any minicomputer comparable with the Digital Equipment Corporation's PDP-11/45 minicomputer which is available in the RADC laboratories (the IBM system operates on the small

IBM System-7) (Ref. 1).

The data extraction mechanism would have to be able to input the serial digital signal space data from the modem output port for the external Eye Pattern Generator, convert it to two words of signal space coordinates, and feed the coordinates to the minicomputer for processing. As in the external Eye Pattern Generator (Fig. 3), the data would have to be demultiplexed and converted to two words of parallel data. As shown in Figure 5 the extraction mechanism would contain input buffers, a demultiplexer, two serial to parallel converters, plus output buffers and handshaking circuits to match with the computer interrupt protocol. A detailed design was not developed in this thesis because the working design would incorporate several mechanisms already available in the RADC laboratories which could be more efficiently interfaced by the lab technicians who are familiar with them.

The minicomputer which would be utilized for signal space analysis and display would have to have an interrupt capability or possibly a direct memory access capability (DMA). The interrupt or DMA capability would allow the signal space coordinates to be fed into the computer as they would be received in the data extraction/interface device one point at a time. At the highest data rate of 2400 baud, there would be 416 microseconds between data points. Therefore the interrupt handler of the minicomputer would have to complete the servicing routine in that time frame or

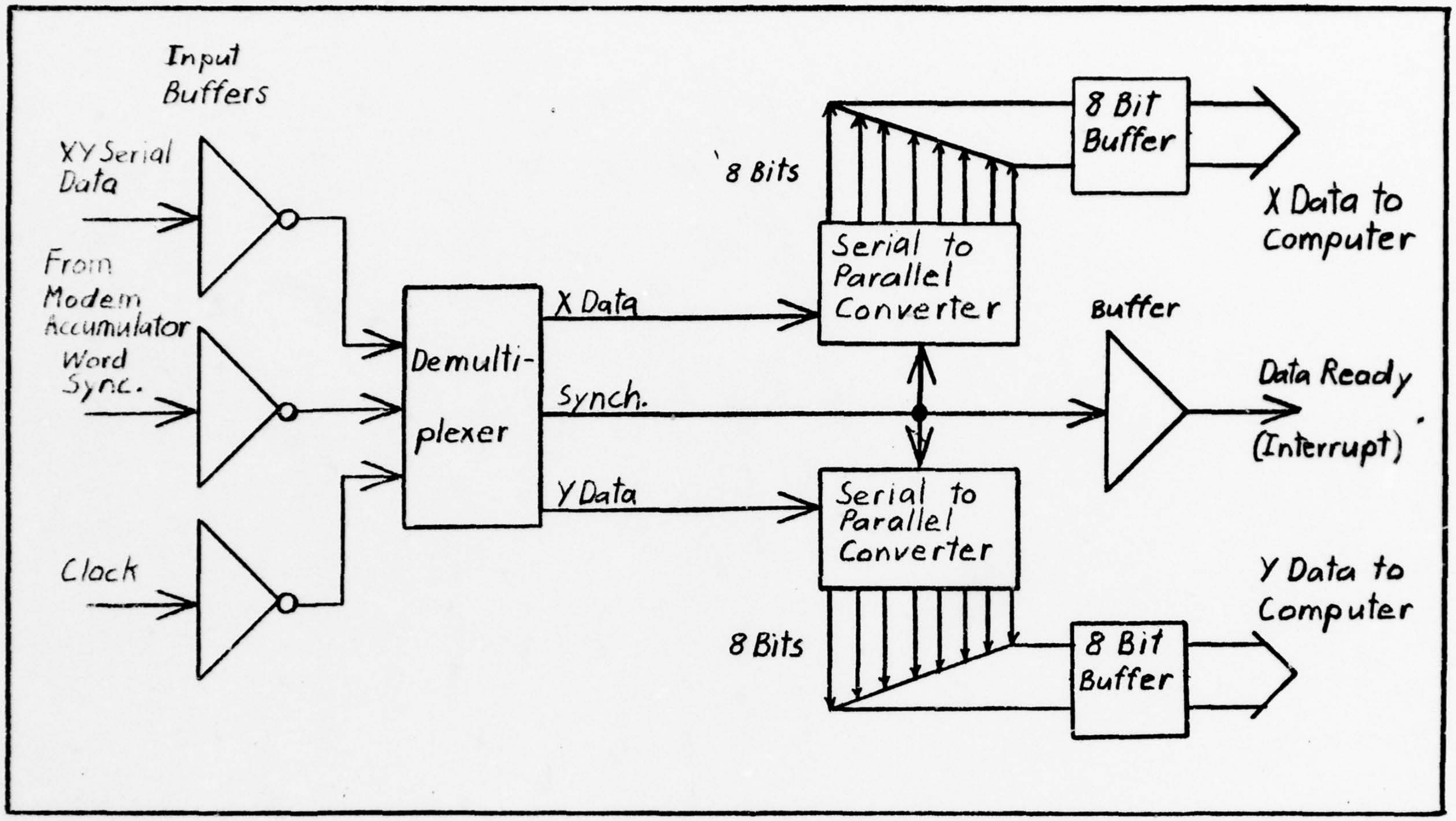


Figure 5. Conceptual Schematic of Signal Space Pattern Data Extraction Circuit (Interface Between Modem and Minicomputer)

subsequent data points would be lost. The worst case time for a PDP-11/45 minicomputer to acknowledge and branch to the first instruction of an interrupt servicing routine would be 17.19 microseconds. The basic move of the input data from the external transfer register to a location in the memory would only be required once and the worst case time would be 4.28 microseconds. Therefore, there would be 394.53 microseconds left over for the basic operations of the interrupt handler routine and indexing of the input buffer stack pointer (Ref. 5). These time frames would be adequate for the simpler current data analysis system developed in this thesis since the subsequant data points after the initial filling of the input buffer would not be needed. However, for an expanded analysis system, these time frames would be critical and require that more elaborate input buffering techniques be developed. DMA would be considerably faster, but a more complex interface would be required between the modem and computer (Ref. 6).

The output of the analysis system would consist of graphical displays showing the desired quantitative information on the line perturbations. These displays could be shown on a video screen or printed out in hard copy. The FORTRAN program included in this thesis was designed to print the displays on the line printer.

Development of Program Structure

The development of the concepts and equipment requirements for real time signal space analysis were straight forward, however the development of the computer program which would perform the analysis was a much more difficult procedure. The program had to be capable of converting two integer values of x and y into parameters which would describe the transmission line condition in standard terms and it had to be done in such a way that the program could be expanded at a later date to include the capability of providing historical data on the transmission line performance. Therefore, it was decided to use design techniques which were developed in a previous AFIT thesis (Ref. 3) to define and develop the program requirements and parameters. Also, these techniques would allow the program to be expanded and adapted to different computer systems at a later time much easier than if just a single real time analysis program were developed for a particular machine. The development of the program structure first consisted of defining all of the requirements and parameters of the IBM-LQM system so that the entire scope of an expanded system could be included in the basic design. Then a structured programming technique was applied which allowed the development of the diagrams in Appendix A. These diagrams were then converted into a basic structure chart as described in Reference 3. The basic structure chart was then refined so that it would make possible the straight forward programming of the real time analysis or current data processing sections of the overall system. Since only the definition phase and the final program structure were peculiar to this thesis, the refinement phases were not developed in the thesis text.

Definition of Program Requirements and Parameters. The program input, output, and control requirements and parameters were developed through an iterative process of progressively increasing the detail of functional diagrams. Three levels of these diagrams are shown in Appendix A, but only the last level is explained here as it contains all of the program requirements and parameters of a complete system.

The data input would consist of the two values of x and y which describe a single point in the signal space (for the Codex 9600 LSI modem, these two words have eight bits each and therefore have integer values between plus or minus 128). The data input for a complete system would be started at time zero and run continuously with x and y values being received simultaneously at regular rates of 2400 or 1600 baud, depending on the data rate of the modem. However, the data rate would not affect the operation of the program since it is designed for use at the highest rate. It should be noted here, that the current data processing elements would not need to operate continuously if they were all that were to be developed for the analysis system. The data input would simply be started when the operator requested, and the input routine would only run until an input buffer was filled.

The output of the program would consist of displays which show standard parameters of transmission lines. It was decided that the displays developed by the IBM-LQM system provided all of the needed transmission line parameters, so the output displays of this program structure were patterened after the IBM displays. The displays would consist of real time displays showing the signal space, the amplitude and phase deviations, and the phase deviation frequency spectrum. There would also be displays which would show summaries of five minutes, one hour, twenty four hours of a single transmission line (the IBM-LQM system can monitor up to seven lines), and a complete summary of all lines for a twenty four hour period. These summaries would contain numerical averages and standard deviations for amplitude deviation, phase deviation, phase jitter, random noise, phase dispersion, and line quality. The summaries would also contain the number of severe or medium phase and amplitude hits for the particular period of the summary. Though the summaries would add a great deal of information capability to the analysis system, they would also require several thousand man hours to be developed (Ref. 1). Therefore they are only described here to show that they were considered in the basic design, but not with the intent of being developed in this thesis. The displays could either be shown on a video screen or be printed on a hard copy.

To convert the input from xy data to quantitative output displays, several operations would have to be performed on each data point and on different groupings of data points as they would be sequentially received. For the current data displays, the xy coordinates would need to be converted to polar coordinates, and the deviations from the signal space reference points or target points would need to be determined in polar coordinates. Then the phase deviations would have to be fast Fourier transformed to determine the phase deviation frequency spectrum or phase jitter frequencies. Finally, the data would need to be arrayed so that it could be displayed in graphical form. If the summary displays and historical capabilities were to be added to the system, then the current processed data, the original raw xy data, plus additional statistical processed data would have to be processed, arrayed, and stored. To obtain the final configuration of the functional diagrams as shown in Appendix A and of the program structure diagrams as shown in Figure 6 through 10, several iterations of design were required. These iterations included the use of bubble charts and several different program structure charts, however, it was decided that complete explanations of the iterations were not required to understand the final program structure.

Program Structure Diagrams. The top level program structure diagram (Fig. 6) is arranged with the input or efferent element on the left and with the output or afferent element on the right so that data flow and order of processing operations would tend to be left to right. Each of the

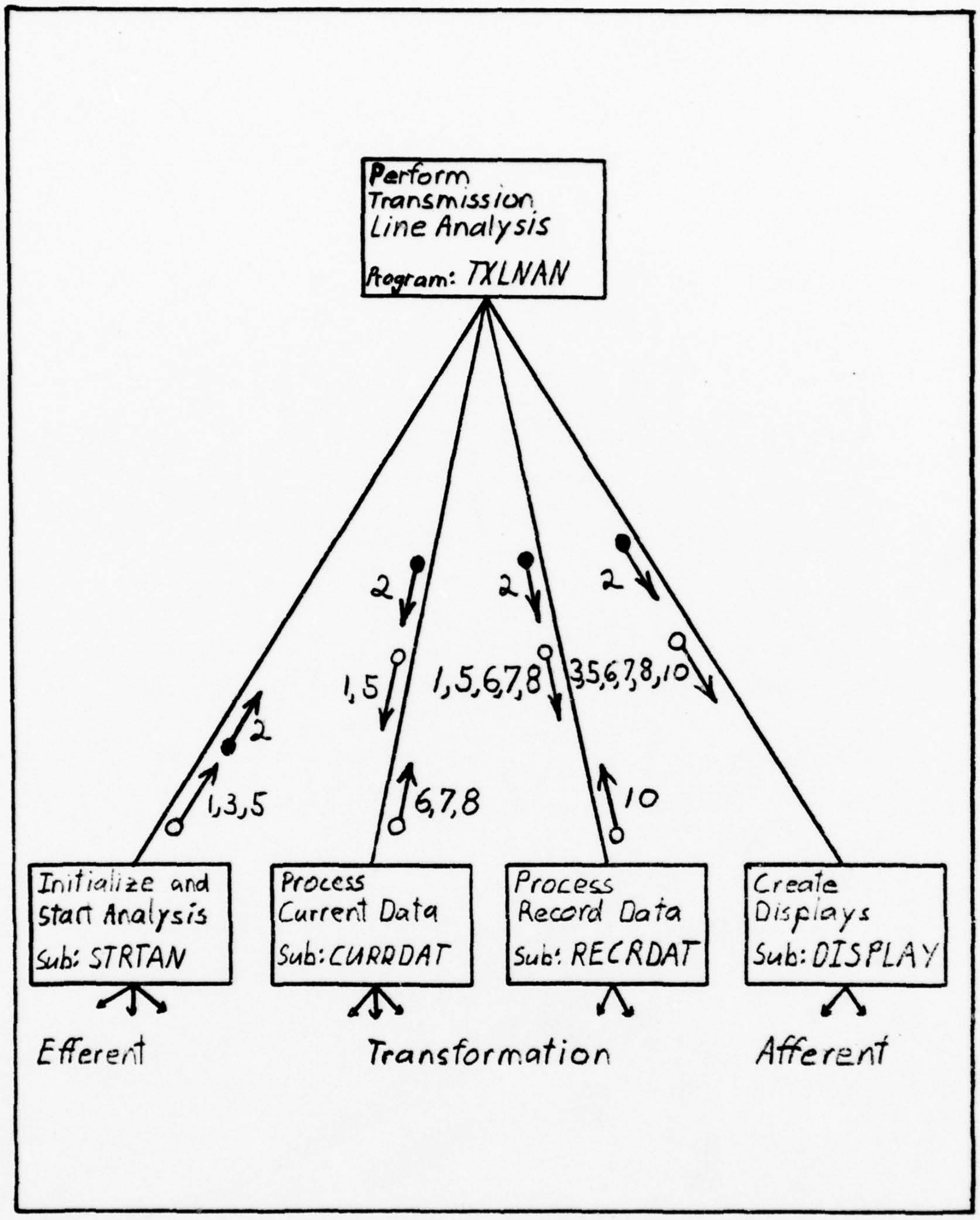


Figure 6. Top Level of Program Structure Diagram

#	DESCRIPTION	FROM	TO
1.	Initialization Parameters	ENTPARM	STRTAN
		STRTAN	TXLNAN
		TXLNAN	CURRDAT
			RECRDAT
2.	Constants for Target Coordinates and Timing Control	ENTPARM	STRTAN
		STRTAN	TXLNAN
		TXLNAN	CURRDAT
			RECRDAT
			DISPLAY
		CURRDAT	APDEV
		RECRDAT	COMPRES
			HISTORY
		DISPLAY	SELDAT
		SELDAT	SLCURR
			SLFMSM
			SLLNSM
			SLHRSM
			SLDASM
3.	External Control Data (Such as Display Selection Requests)	CONINT	WAIT
		WAIT	STRTAN
		STRTAN	TXLNAN
		TXLNAN	DISPLAY
		DISPLAY	SELDAT
			SELDIS
		SELDAT	SLCURR
			SLFMSM
			SLLNSM
			SLHRSM
			SLDASM
		SELDIS	DISXY
			DISAPD
			DISPDFS
			DISFMSM
			DISLNSM
			DISHRSM
			DISDASM

Table 1. Data and Control Identifiers for Program Structure Diagrams (Figure 6 through Figure 10)

#	DESCRIPTION	FROM	TO
4.	XY Coordinates of Single Received Data Point	DATINT	WAIT
		WAIT	DATRDY
5.	Full Buffer of 120 Points of XY Data Coordinates	DATRDY	STRTAN
		STRTAN	TXLNAN
		TXLNAN	CURRDAT
			RECRDAT
			DISPLAY
		CURRDAT	XYTOPOL
		RECRDAT	COMPRES
		DISPLAY	SELDAT
		SELDAT	SLCURR
6.	Amplitude and Phase Data Coordinates for 120 Received Data Points	XYTOPOL	CURRDAT
		CURRDAT	APDEV
			TXLNAN
		TXLNAN	RECRDAT
			DISPLAY
		RECRDAT	COMPRES
		DISPLAY	SELDAT
		SELDAT	SLCURR
7.	Amplitude and Phase Deviation Data of 120 Received Points from the Nearest Target Point	APDEV	CURRDAT
		CURRDAT	FRQSPEC
			TXLNAN
		TXLNAN	RECRDAT
			DISPLAY
		RECRDAT	COMPRES
		DISPLAY	SELDAT
		SELDAT	SLCURR
8.	Phase Deviation Frequency Data for 120 Received Points	FRQSPEC	CURRDAT
		CURRDAT	TXLNAN
		TXLNAN	RECRDAT
			DISPLAY
		RECRDAT	COMPRES
		DISPLAY	SELDAT
		SELDAT	SLCURR

Table 1 Continued. Data and Control Identifiers for Program Structure Diagrams (Figure 6 through Figure 10)

#	DESCRIPTION	FROM	TO
9.	Statistical Data (Compressed Data)	COMPRES	RECRDAT
		RECRDAT	HISTORY
10.	Catalogued and Line, Date, Time Referenced Summary Data	HISTORY	RECRDAT
		RECRDAT	TXLNAN
		TXLNAN	DISPLAY
		DISPLAY	SELDAT
		SELDAT	SLFMSM
			SLLNSM
			SLHRSM
			SLDASM
11.	Plot Data for 120 Values of Current Data Display	SLCURR	SELDAT
		SELDAT	DISPLAY
		DISPLAY	SELDIS
		SELDIS	DISXY
			DISAPD
			DISPDFS
12.	Plot Data for Summary Displays	SLFMSM	
		SLLNSM	
		SLHRSM	
		SLDASM	SELDAT
		SELDAT	DISPLAY
		DISPLAY	SELDIS
		SELDIS	DISFMSM
			DISLNSM
			DISHRSM
			DISDASM

Table 1 Continued. Data and Control Identifiers for Program Structure Diagrams (Figure 6 through Figure 10)

levels from top to bottom (Fig. 6 to Fig. 10) correspond roughly with the levels of functional diagrams in Appendix A. This arrangement would allow the different functional operations to be separated and ultimately written into separate subroutines.

Master Control (Fig. 6). The top block is the master control and it would determine which type of action at the next level would be performed. The next level of four functions contains the major categories of system activities which are data acquisition and program initialization, real time or current data processing, record data processing, and display output formulation.

Initialization and Analysis Start (Fig. 7). The data acquisition and program initialization block would be the starting point of the analysis. It contains a block for initialization of program constants such as the coordinates of the target points or reference transmitted signal space points, a block for an idle mode when all other processing would have been completed, and a block which would transfer a full input buffer of data into a processing location within the computer to isolate input and processing actions. The initialization or "enter parameters" block would be a one time operation in that it would enter all the needed constants of the program and set all of the decision flags at the program start. The idle or wait block would idle the computer until an interrupt signal was received. This block contains two subdivisions for control interrupts and

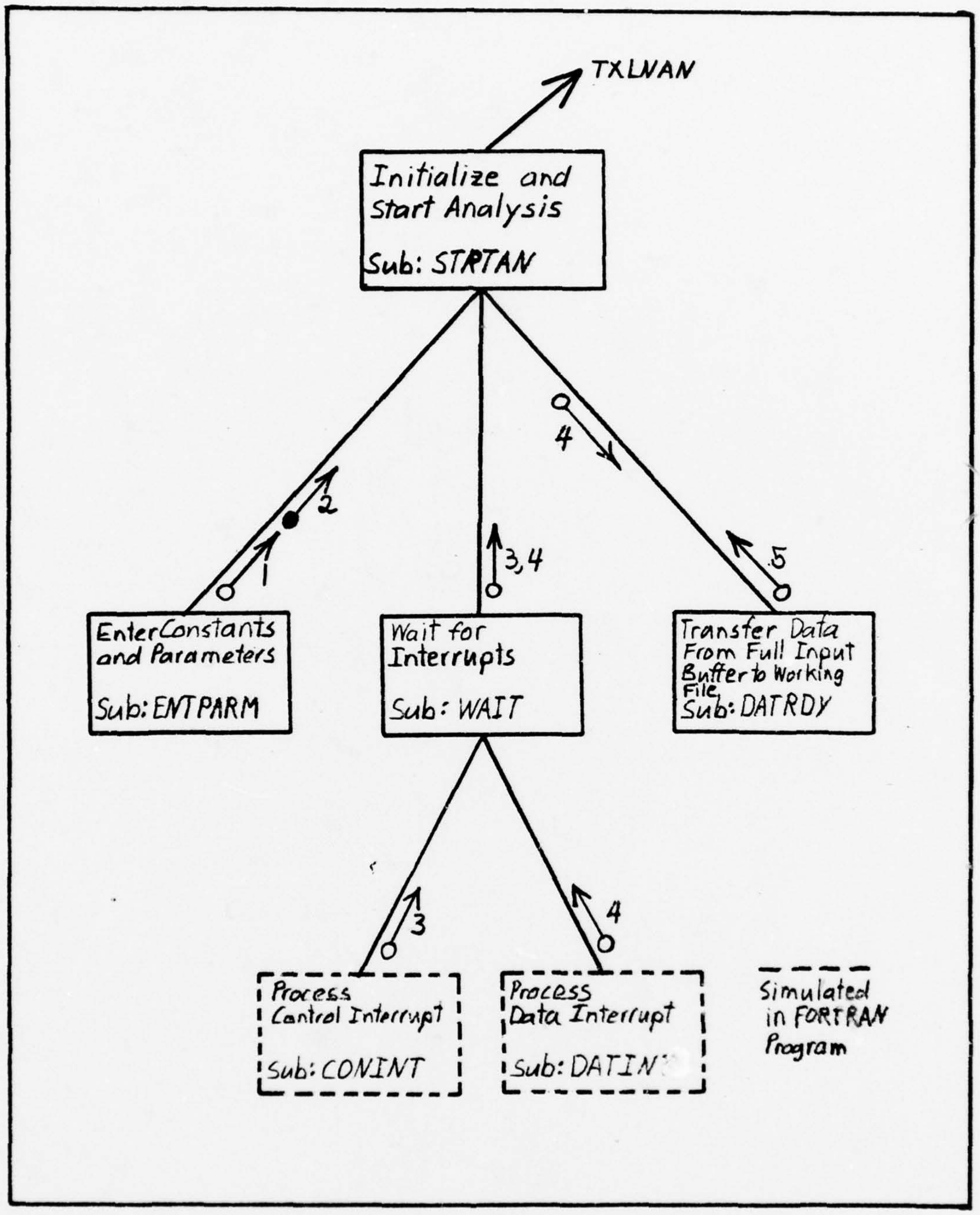

TXLNAN
Initialize and Start Analysis
Sub: STRTAN
4
2
1
3,4
5
Enter Constants and Parameters
Sub: ENTPARM
Wait for Interrupts
Sub: WAIT
Transfer Data From Full Input Buffer to Working File
Sub: DATRDY
3
4
Process Control Interrupt
Sub: CONINT
Process Data Interrupt
Sub: DATINT
Simulated in FORTRAN Program

Figure 7. Efferent Elements of Program Structure Diagram

data interrupts. The control interrupts were included so that the different displays could be requested externally. The data interrupt would allow the two words of x and y coordinates to be read in from the external interface device and put into an input buffer. The input buffer would be necessary in the expanded system so that other processing actions could be accomplished while the buffer was being filled by the data interrupt and so that no data points would be lost during the process. The last block in the start-up of the analysis is the final acquisition of data or data ready block. This block would transfer the whole input buffer of xy data into a working location of memory so that the other processing operations could be performed without interfering with the data input.

<u>Current Data Processing</u> (Fig. 8). The current data processing block would prepare the xy data so that the record data block could average and file the data and so that the display block could create the appropriate displays of the current data. The current data block contains three elements working in series which would convert the cartesian xy coordinates of the signal space to polar coordinates of amplitude and phase, calculate the amplitude and phase deviations from the target or reference transmitted points, and calculate the phase deviation frequency spectrum.

<u>Record Data Processing</u> (Fig. 9). The record data block would transform the current data values of xy coordinates, amplitude and phase deviations, and phase deviation

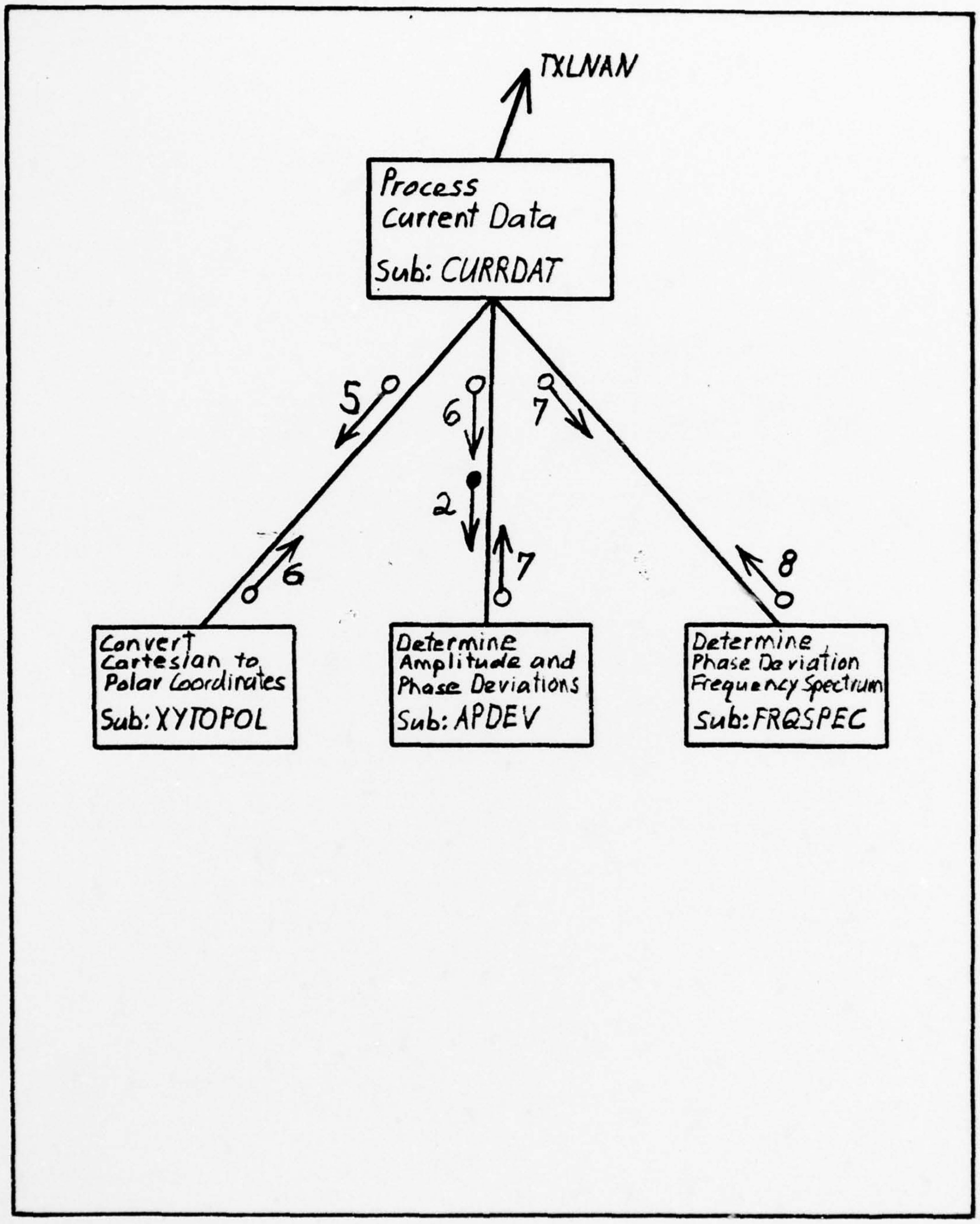

TXLNAN
Process Current Data
Sub: CURRDAT
5
6
7
2
6
7
8
Convert Cartesian to Polar Coordinates
Sub: XYTOPOL
Determine Amplitude and Phase Deviations
Sub: APDEV
Determine Phase Deviation Frequency Spectrum
Sub: FRQSPEC

Figure 8. Current Data Elements of Program Structure Diagram

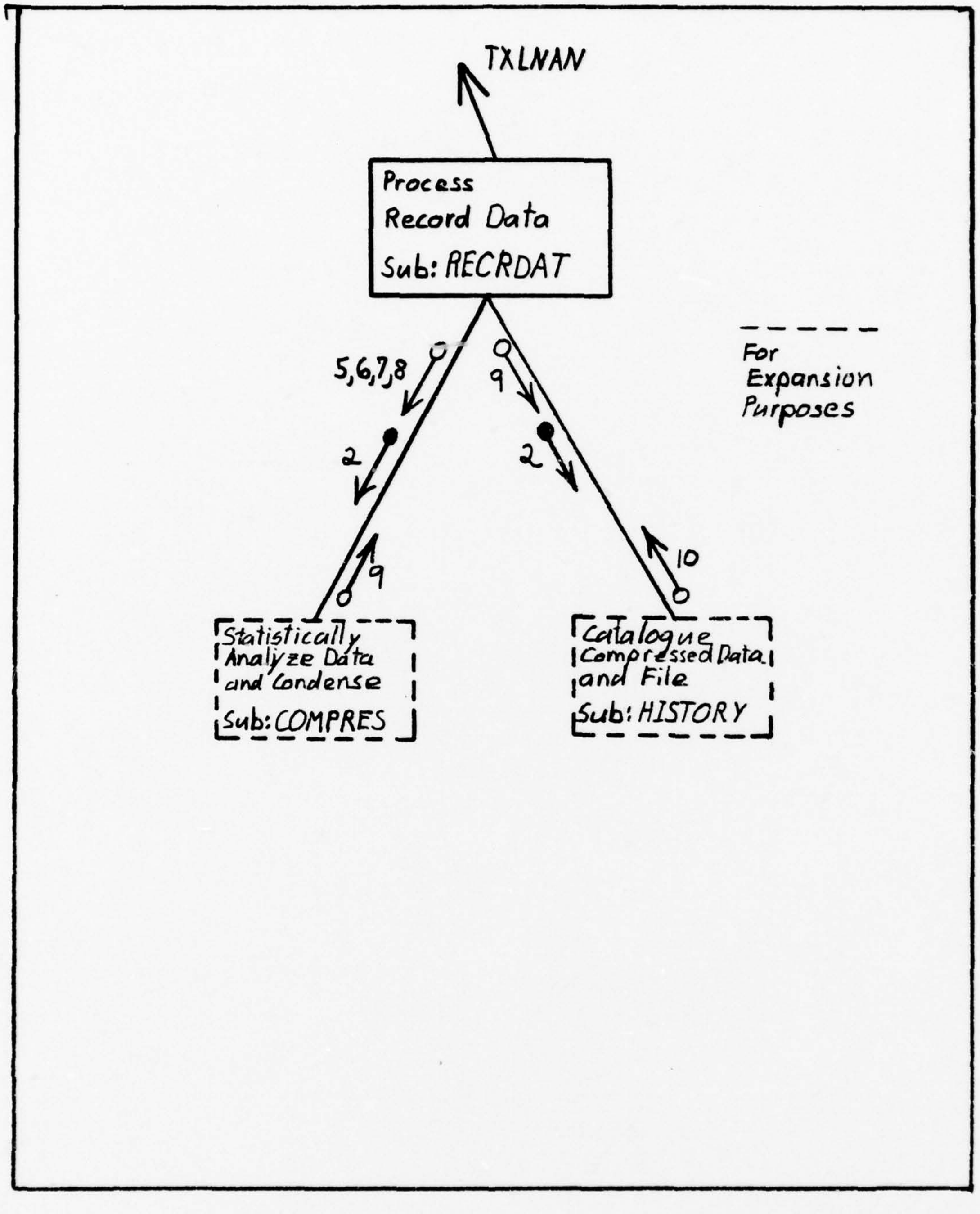


Figure 9. Historical Data Elements of Program Structure Diagram

frequencies into time averages of deviation, jitter, noise, quality, and hits, then it would put them into a permanent file. This block has two functional blocks of data compression and data history which would respectively perform the averaging calculations and perform the permanent cataloging or storing functions. Conceivably, these two blocks could be broken down to several more levels of functions to obtain a more readily programmable system. However, these blocks would be highly complex and would require an extensive study of operations necessary to calculate the summary data values for the historical transmission line analysis. Also, these blocks are not needed for the current data analysis, so they are included only as expansion ports.

Display (Fig. 10). The display block contains two functional blocks of which one would select the appropriate data for a particular display, and of which the other would select the appropriate display routine for the selected data. The data selection block has five modes of selection depending on the display to be created. If the desired display was one of the three current data displays, the selection block would gather all of the current data needed to create all three current data displays since all three would often be requested for comparison purposes. The other selection functions would be tailored to each display. The display selection block would simply call the appropriate display subroutine from the selection of the seven different subroutines in the expanded system, however only three

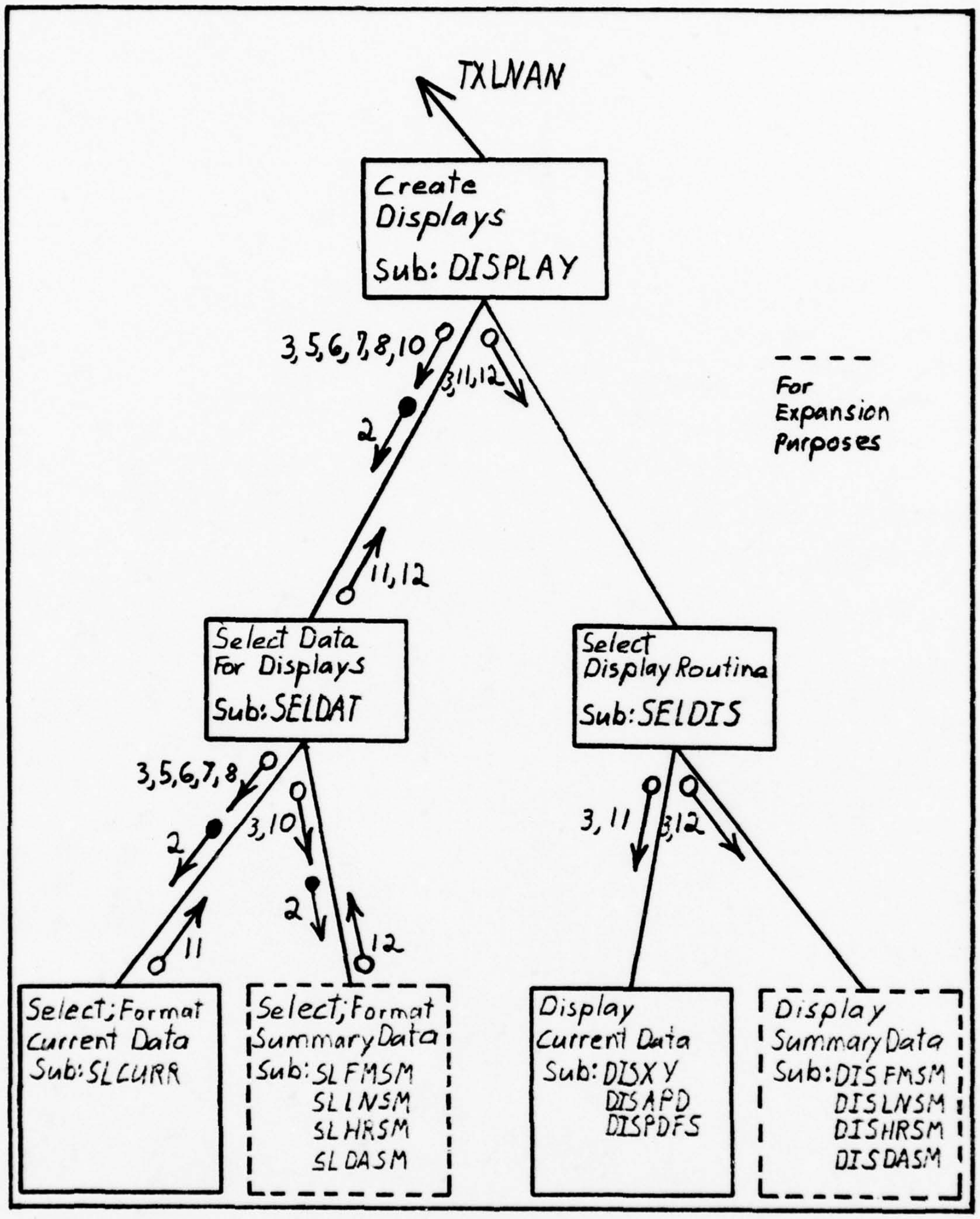


Figure 10. Afferent Elements of Program Structure Diagram

display subroutines are needed for a current data analysis system.

Development of FORTRAN Program

With use of the program structure diagram, the development of a functional FORTRAN program was straight forward. Since the program structure contained all of the necessary aspects and parameters of the system, a complete FORTRAN program was written to correspond directly with each block on the program structure diagram. Though the resulting FORTRAN program is not the fastest for processing time, nor most efficient for memory requirements, it does contain all of the needed operations required in the current data analysis system and therefore can be optimized to fit a particular minicomputer.

The first step in the FORTRAN program development was to assign a subroutine to each functional block. Then the development became a process of writing calling routines for the top two levels of the structure diagram and of writing short functional programs for each specific task to be performed by the remaining blocks. The top most block became the master control program which can call any of the four blocks or subroutines in the next level. These four subroutines in turn can call the appropriate subroutines under themselves to perform the actual computations or transfers of data within the system. The details of the program names, decision logic, flag names, and file names are available in Appendix B and Appendix C where there are flow diagrams and a copy of the FORTRAN program. Reference to these appendices and Figure 6 through 10 may prove helpful in reading the following subroutine developments.

Efferent Subroutines. The efferent elements are comprised of all the subroutines under and including STRTAN (Start Analysis). ENTPARM (Enter Parameters) is simply used one time at the inital start of the program to clear and preset the appropriate flags, and to enter all of the necessary constants. WAIT (idle mode) would normally be a constant test loop for interrupts, but in this program, which was developed for a multiuser computer system, flags are set so that it sequentially selects DATINT (Data Interrupt) and CONINT (Control Interrupt). DATINT would normally be an interrupt routine to read in a single point of xy information, but for this program, it is used as a simulation routine which generates a full input buffer (120 points) of simulated received signal space coordinates. CONINT would also be an interrupt routine to read in external display selection requests. In an expanded system, the program would operate continuously and process all data as the data is received, and automatically print out only a very limited amount of processed data according to instructions which would be set up within the RECRDAT (Record Data) routines. CONINT would be used to request the additional current data or summary printouts. Since a current data analysis system does not require continuous operation of the program, CONINT would be set up as a keyboard servicing routine which would allow an operator to set the appropriate display request flags and the program start flags. But in this program, CONINT is used as a routine which internally sets the flags to have the current data displays printed out. DATRDY (Data Ready) is a subroutine that transfers an input buffer, when it is filled with 120 points, into another memory location where the xy data can be processed by other subroutines without affecting the input buffer. In the expanded system, this subroutine would be quite critical and timing would have to be carefully analyzed to insure that data would not be lost. However in the current data analysis system, the subsequent data points after the initial filling of the input buffer are not important and therefore the DATRDY timing is not critical.

Transformation Subroutines. The transformation routines include CURRDAT (Current Data), RECRDAT (Record Data) and all of the routines under them. XYTOPOL (XY to Polar) transforms 120 points of xy cartesian coordinates into 120 points of polar coordinates (amplitude and phase). APDEV (Amplitude and Phase Deviation) determines the amplitude and phase deviations of each received signal space point from its respective reference or target point in the signal space. FRQSPEC (Frequency Spectrum) determines the frequencies of the phase deviations. It is a FFT (fast Fourier transform) program (Ref. 4) which processes 120 points with a 128 point FFT. The 8 additional points in the transform are filled with zeroes. COMPRES (Data Compression) and HISTORY (Permanent Data Storage) are routines intended only for future expansion purposes. The COMPRES routines would consist of complex statistics routines to compress the data by determining averages and standard deviations of amplitude and phase deviations, phase jitter, phase dispersion, noise, number of phase and amplitude hits, and line quality. The HISTORY routine would prepare the compressed data for permanent storage with time and line references for cataloging purposes, then store the cataloged data in permanent files. These two routines were not developed in this thesis as they are not required for the current data analysis and display.

Afferent Subroutines. The afferent elements are comprised of the subroutines under and including DISPLAY. SELDAT (Select Data) and SELDIS (Select Display) are selection routines which determine through the status of control flags the data routines and display routines which are to be executed. The SLCURR (Select Current Data) retrieves all of the current data required to plot the three current data displays of signal space, amplitude and phase deviation, and phase deviation frequency spectrum and it puts the data into a file which can be used by each of the current data display routines. The SLFMSM (Select Five Minute Summary Data), SLFLNSM (Select Line Summary Data), SLHRSM (Select Hour Summary Data), and SLDASM (Select Day Summary Data) routines are only routines included for expansion if the RECRDAT routines are developed. These four selection routines would obviously have to fit in with the cataloging procedures in the development of the HISTORY routine. The three routines, DISXY (Display Signal Space), DISAPD (Display Amplitude and Phase Deviations), and DISPDFS (Display Phase Deviation Frequency Spectrum), are the line printer routines which put the appropriate current data retrieved by SLCURR into a formatted output, then direct a printout of the current data displays. The remaining five summary display routines are again only intended for use with system expansion of RECRDAT routines. They would format the retrieved HISTORY data and direct a line printer or video output.

Operations Concepts. The FORTRAN program in Appendix C and an actual working program will be somewhat different. The FORTRAN program was designed for use as a simulation program on a multiuser computer system. Therefore, it starts, produces its own data, then executes the analysis routines. It starts with TXLNAN, initializes with ENTPARM, generates one input buffer of 120 points of xy data in the DATINT subroutine, and transfers the data in the DATRDY subroutine. Then the program processes the 120 points in the CURRDAT, XYTOPOL, APDEV, and FRQSPEC subroutines, and returns to CONINT. Here it reads in the display requests (sets appropriate flags), then performs DISPLAY, SELDAT, SELDIS, DISXY, DISPDFS, and finally stops. In a real system, the program would read in a data point in the DATINT interrupt routine, then sit at WAIT until another interrupt for DATINT was received. When 120 points would be received, the program would continue as in the previous description. If the system would ever be developed into the expanded system, then the program would have to run continuously. Instead of stopping, it would just jump back to the WAIT mode and begin the cycle again. However, while all the processing of the 120 points was being accomplished, the DATINT interrupt routine would still be bringing in new data points so all of the other processing would have to cycle through and back to WAIT within 50,000 microseconds in order to catch the next buffer of points. A great deal of timing considerations and advanced programming techniques would be required to successfully expand the system into an actual working system.

IV. Discussion of Results

Though the entire analysis system was not constructed and the analysis program was only simulated, there are sufficient results from the simulation to show that the current data analysis system will provide quantitative information on transmission line perturbations. The three current data displays of signal space pattern, amplitude and phase deviations, and phase deviation frequency spectrum can be used to observe and measure several of the line parameters for perturbations which cannot be measured on the oscilloscope eye pattern display. Though an oscilloscope with a "freeze" or memory capability can be used to obtain displays similar to the computer current data signal space display, it still will not have the precision of the computer display. The average values of deviations, noise, phase dispersion, number of hits, and phase jitter are not directly available through the three current data displays, but the exact instantaneous values of deviations and phase jitter are directly shown on the displays. These displays can be used to roughly see the average values of deviations and jitter, and they can also be used to determine noise levels if reference displays are available for comparison.

Simulation of Line Perturbations

The generation of simulated data was accomplished within the DATINT subroutine. The procedure consisted of a

DO loop which cycled 120 times to fill the input buffer with 120 xy signal space points as if they had been serially received. Within the DO loop, there was a uniform random selection of one of the 16 target points (which were contained in memory in both polar and cartesian coordinates), and then amplitude and phase deviations were added to or subtracted from the coordinates of the target point to obtain a simulated received data point in polar coordinates. Then, while still in the DO loop, the coordinates are switched to cartesian integer values between plus or minus 128 for entry into the input buffer. The two equations used to generate the amplitude and phase coordinates in the DO loop prior to conversion to cartesian coordinates were:

```
A = TARGET*REAL(APT(IT))+AD+AH                              (1)
P = AIMAG(APT(IT))+PJAMP*COS(2*PI*I/PJER)+PD*NPD+PH         (2)
```

where:

A = amplitude of simulated received data points

P = phase of simulated data point

TARGET = 0 or 1 multiplier depending on whether or not a lineout is desired

REAL(APT(IT)) = a uniform random target point amplitude value

AIMAG(APT(IT)) = the same uniform random target point phase value

PJAMP = phase jitter amplitude

PJER = phase jitter period (I = DO loop iteration)

AD = amplitude deviation (normal gaussian random variable with a standard deviation of SIGMA)

PD = phase deviation (normal gaussian random variable with a standard deviation of SIGMA)

NPD = 0 or 1 multiplier to include or delete random phase deviations

AH = constant amplitude hit deviation

PH = constant phase hit deviation

By appropriately changing the values of the above constants, the line perturbations of lineout, normal gaussian noise, normal gaussian noise on the amplitude with pure jitter on the phase, normal gaussian noise on the amplitude and phase with phase jitter added, and phase and amplitude hits can be simulated.

Quantitative Value of Current Data Displays

Different sets of simulation variables were used to generate a set of displays for each of the six line perturbations previously described. Comparison of the computer current data signal space displays with oscilloscope displays shown in Figure 4 correspond quite well. Also, comparisons between the computer displays themselves show that several different quantitative values of the line perturbations are readily available.

<u>Transmission</u> <u>Lineout</u>. The lineout condition was simulated within the DATINT subroutine by using data values:

SIGMA = 3.0

PJPER = 1.0

PJAMP = 0.0

TARGET = 0.0

NPD = 1.0

AH = 0.0

PH = 0.0

These values cause Equation 1 to provide Normal Gaussian random variables with a standard deviation of 3.0 for a received amplitude and cause Equation 2 to provide uniformly distributed phase angles between plus or minus 180 degrees.

The resulting signal space pattern display in Figure 11A corresponds directly with the oscilloscope display shown in Figure 4. The "T" marks shown in the computer display mark where the reference or target signal space coordinate points are located and the "*" marks denote the actual received coordinate points. As expected, they are all located near the origin.

Since the amplitude and phase deviations are immaterial during a lineout, the FORTRAN program is set to identify any received amplitude of less than 12 as a lineout condition. The amplitude and phase deviation display in Figure 11B reflects in the lineout condition.

The phase deviation frequency spectrum (Fig. 11C) is just the absolute value of a Sinc function since the phase deviations appear as a pulse due to their uniform random distribution.

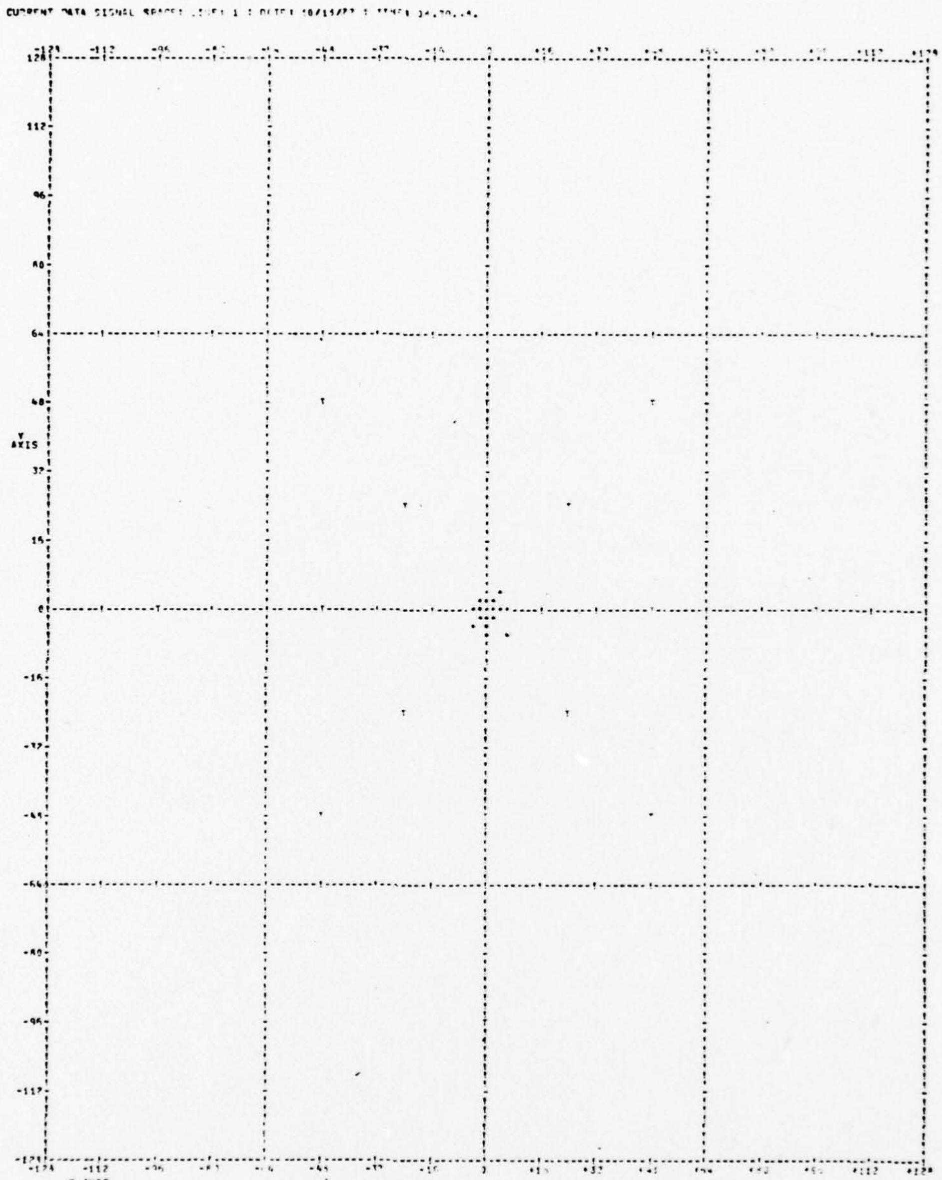


Figure 11A. Current Data Display: Lineout Condition (Signal Space Pattern)

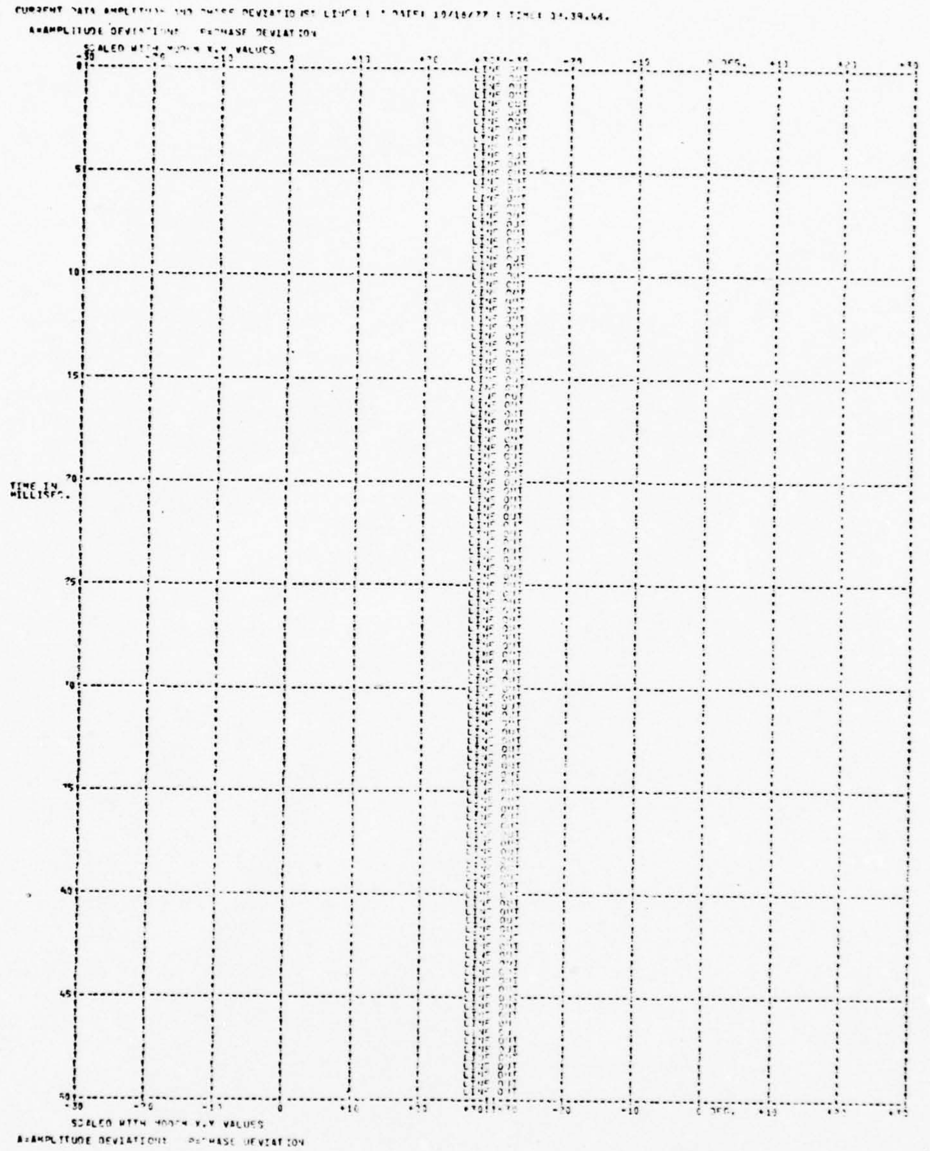


Figure 11B. Current Data Display: Lineout Condition (Amplitude and Phase Deviations)

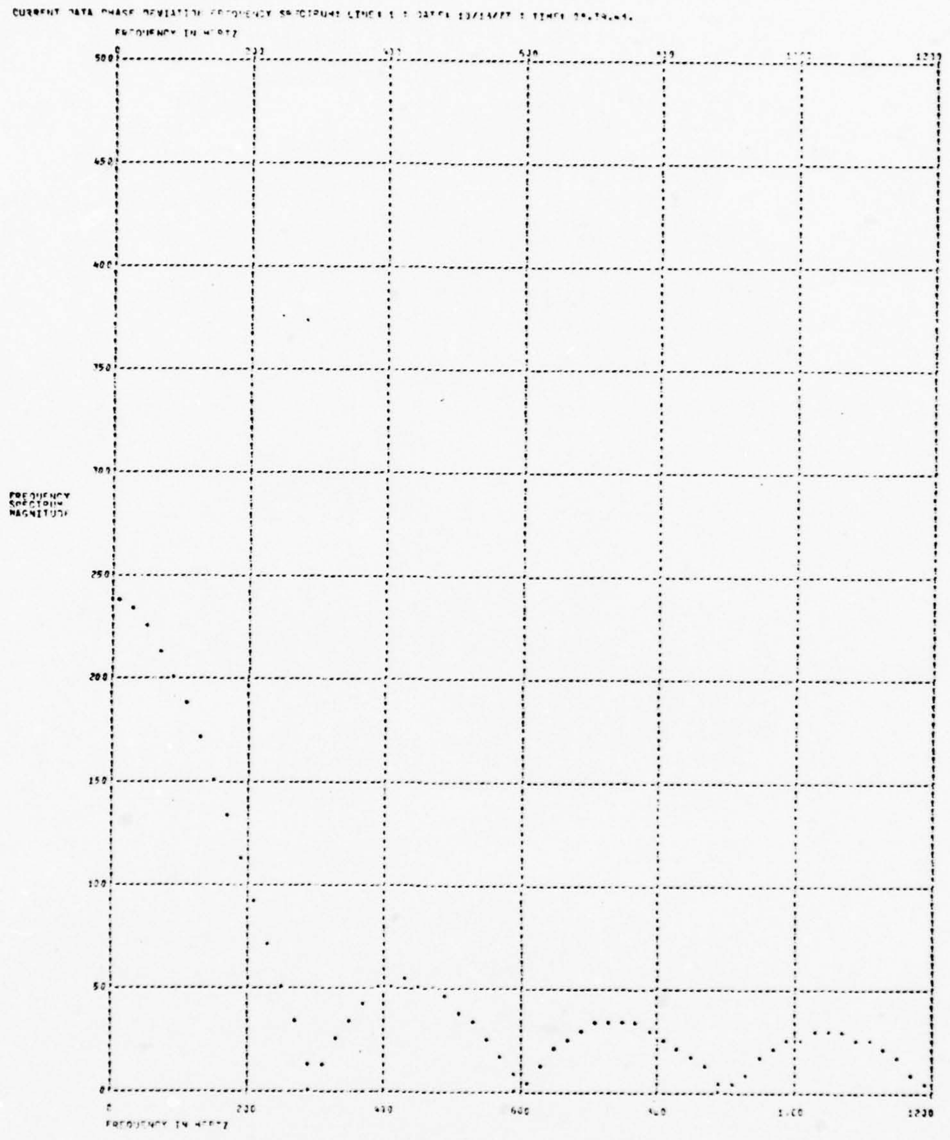


Figure 11C. Current Data Display: Lineout Condition (Phase Deviation Frequency Spectrum)

Normal Gaussian Noise on Amplitude and Phase. This noise condition was simulated within the DATINT subroutine by using data values:

SIGMA = 3.0

PJPER = 1.0

PJAMP = 0.0

TARGET = 1.0

NPD = 1.0

AH = 0.0

PH = 0.0

These values cause Equations 1 and 2 to provide amplitude and phase values which are uniformly distributed between the 16 target points with normal gaussian distributions of amplitude and phase deviations about each respective target point with a variance of 3.0.

The signal space display in Figure 12A corresponds closely with the oscilloscope display in Figure 4 except that the received computer display points are more distinct and the reference target points ("T" points) are provided for more clarity.

The amplitude and phase deviations display in Figure 12B definitely shows an observer more about the nature of the noise and just the signal space display as the average values and the variances of the deviations are roughly available just through observation. If different displays for various levels of known noise can be produced, then these could be used as standards for comparison and actual values of noise could be determined for operational transmission lines.

The phase deviation frequency in Figure 12C shows the frequency distribution is randomly distributed between 0 and 1200 Hz with no dominant frequencies.

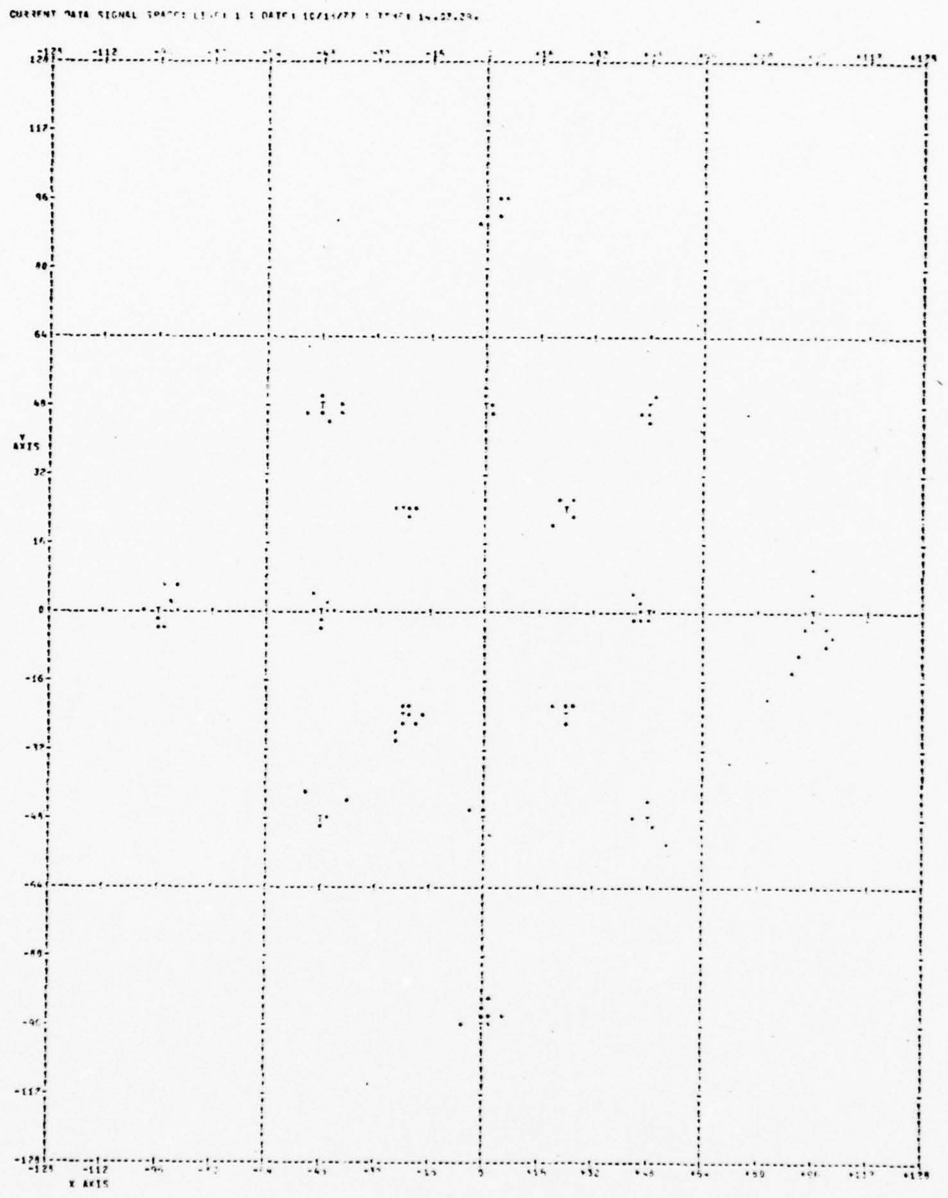


Figure 12A. Current Data Display: Normal Gaussian Noise Condition (Signal Space Pattern)

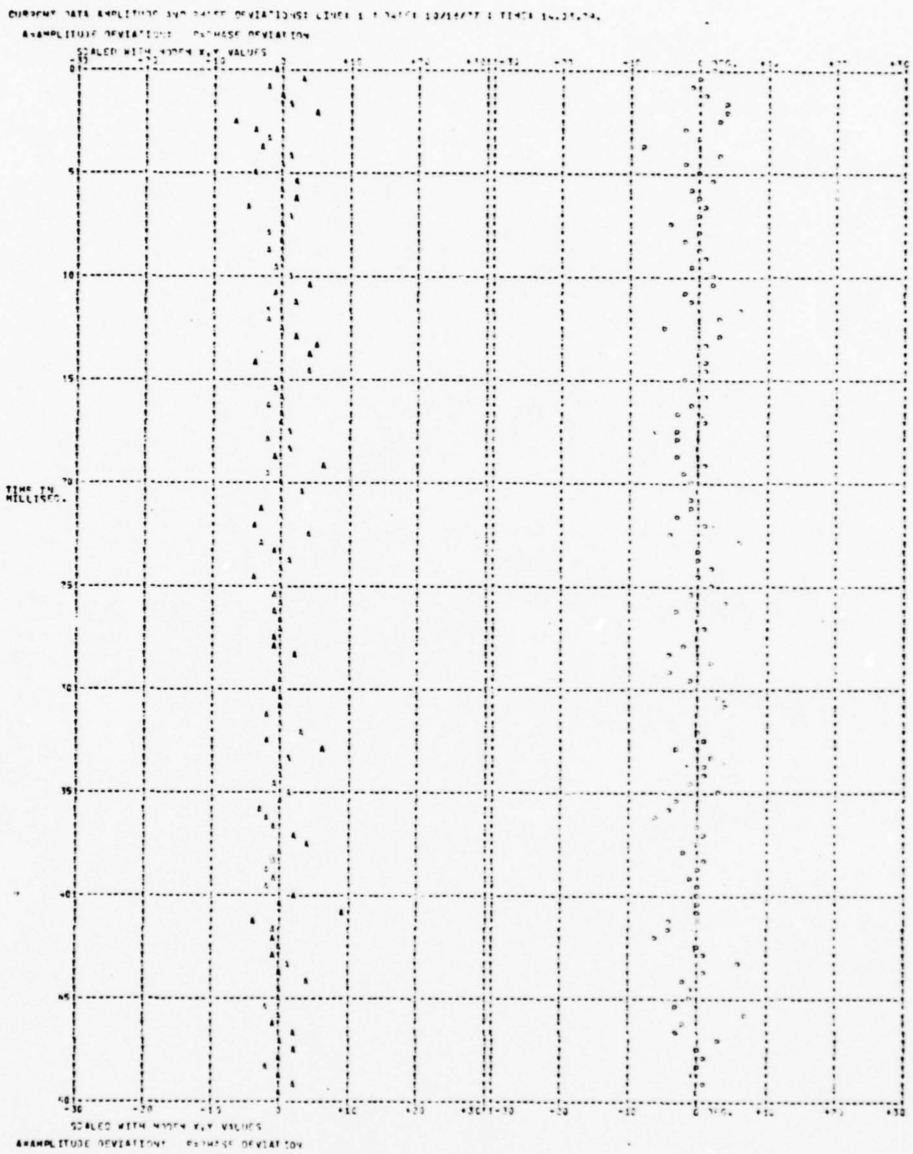


Figure 12B. Current Data Display: Normal Gaussian Noise Condition (Amplitude and Phase Deviations)

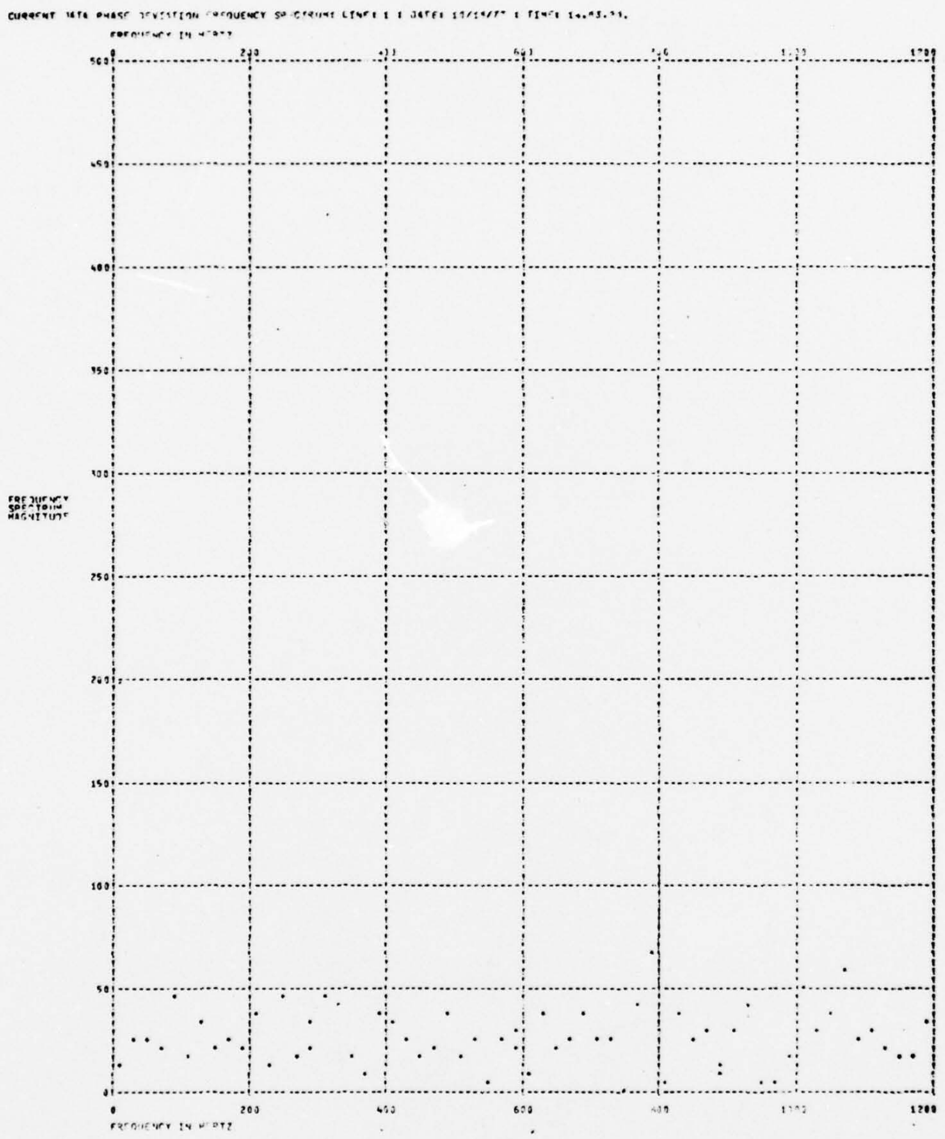


Figure 12C. Current Data Display: Normal Gaussian Noise Condition (Phase Deviation Frequency Spectrum)

Normal Gaussian Noise on the Amplitude; Pure Jitter on the Phase. The noisy amplitude and pure phase jitter conditions were simulated within the DATINT subroutine by using data values:

SIGMA =	3.0
PJPER =	24.0
PJAMP =	10.0
TARGET =	1.0
NPD =	0.0
AH =	0.0
PH =	0.0

These values cause Equation 1 to provide amplitudes with noise as in the previous case, but they cause Equation 2 to provide phase angles with a pure phase jitter of amplitude equal to 10 and a period of 24 received points or 100 Hz at 2400 baud.

The computer signal space pattern display in Figure 13A shows roughly that the phase deviations are arced around the target points as in the oscilloscope display in Figure 4. However, as noted earlier, addition of some noise on the amplitude spreads received distribution around the target point in such a way that phase jitter is difficult to see, much less evaluate.

The amplitude and phase deviation display in Figure 13B clearly shows the phase jitter and phase deviation frequency spectrum in Figure 13C shows the principle frequency at 100 Hz.

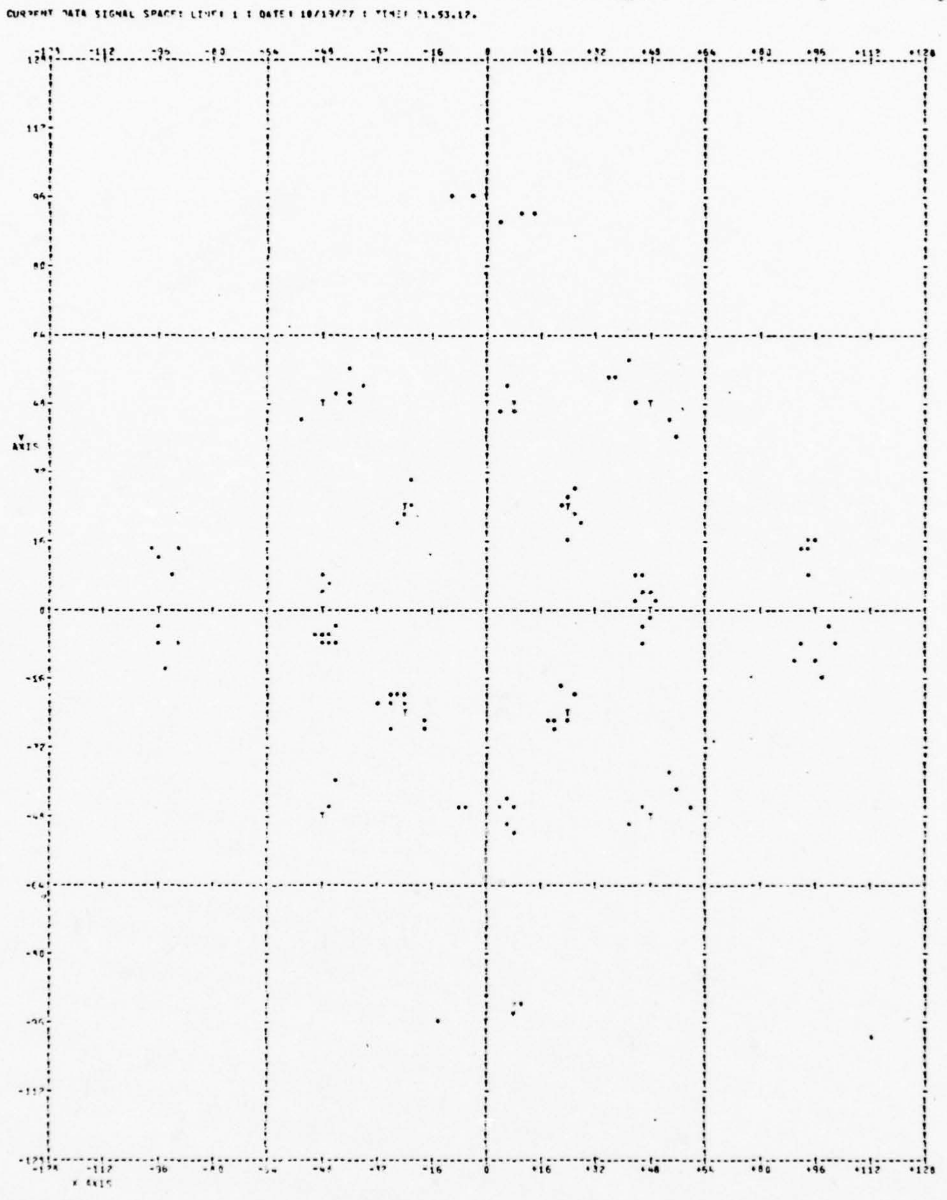


Figure 13A. Current Data Display: Normal Gaussian Noise on the Amplitude with Pure Phase Jitter (Signal Space Pattern)

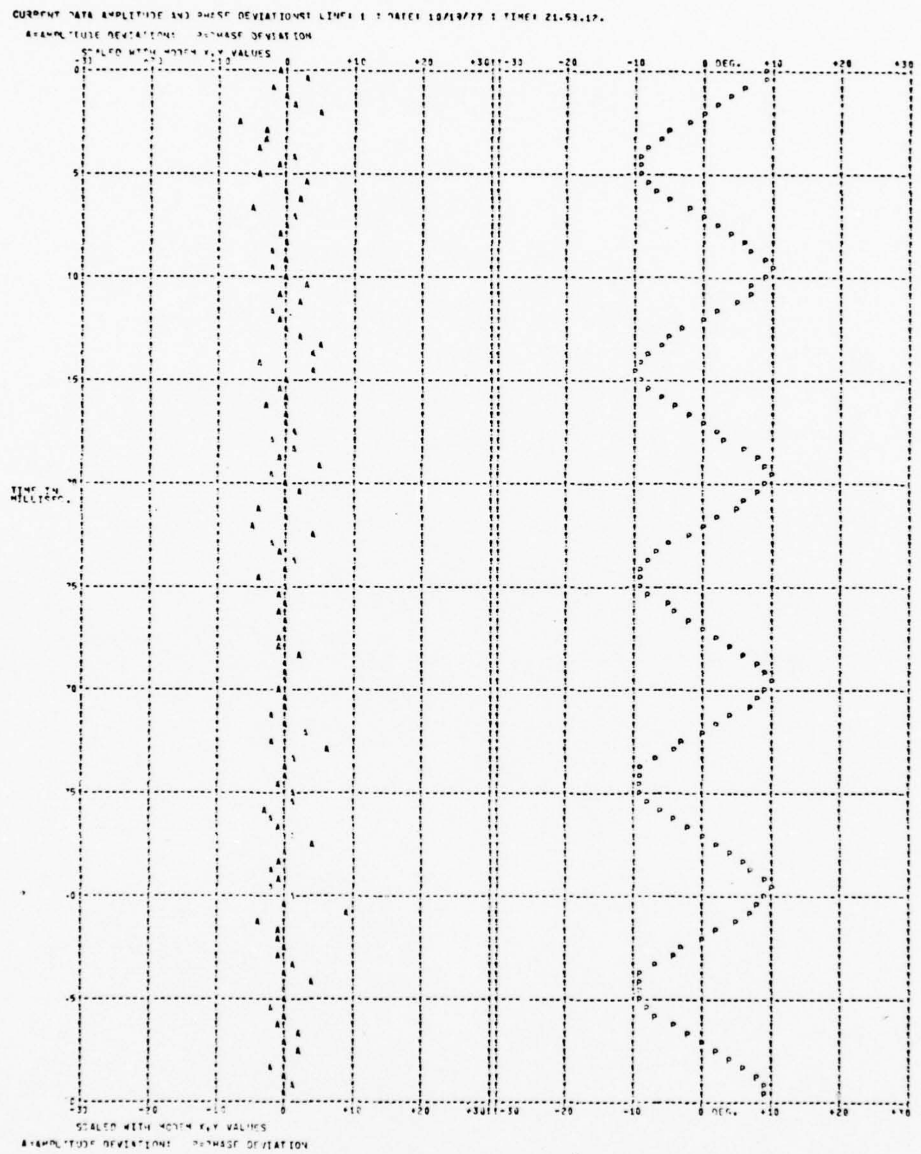


Figure 13B. Current Data Display: Normal Gaussian Noise on the Amplitude with Pure Phase Jitter (Amplitude and Phase Deviations)

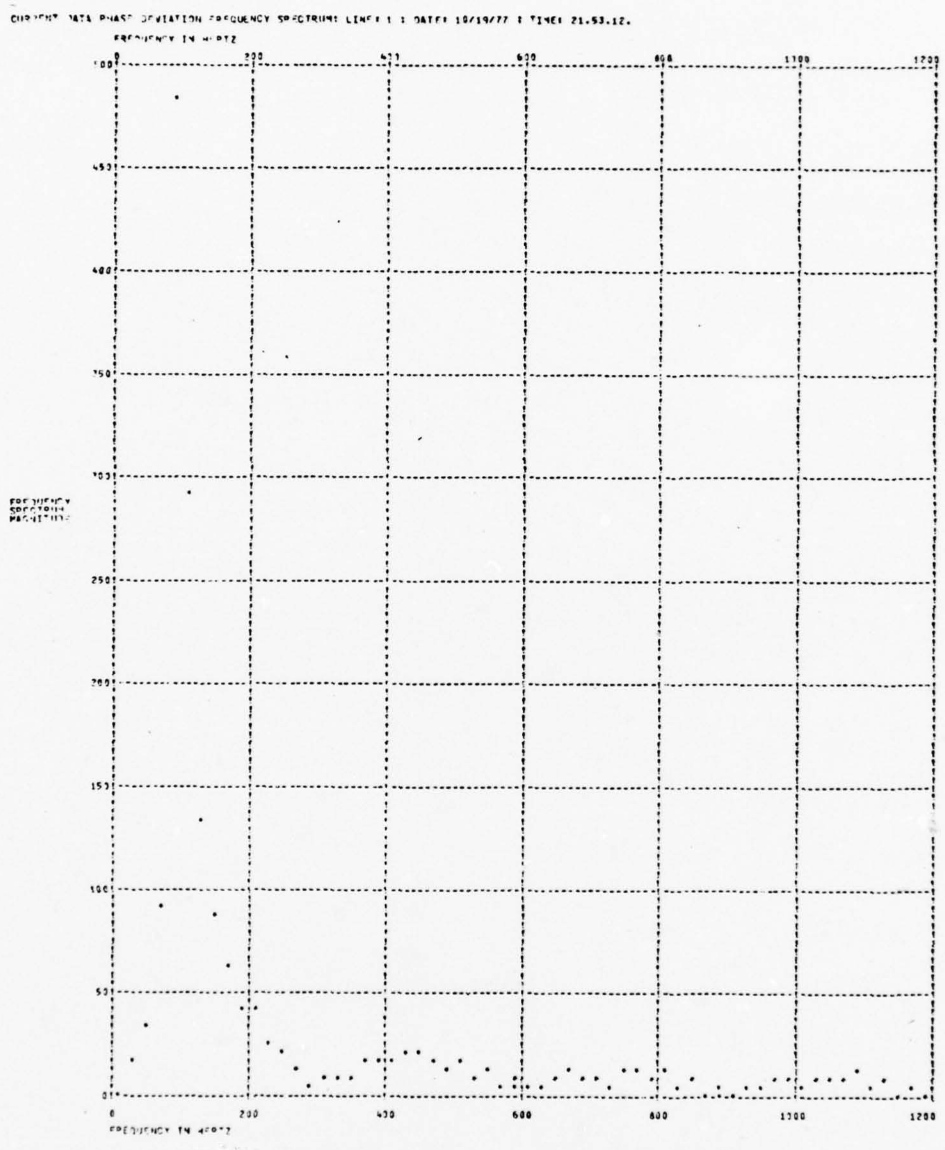


Figure 13C. Current Data Display: Normal Gaussian Noise on the Amplitude and Pure Phase Jitter (Phase Deviation Frequency Spectrum)

Normal Gaussian Noise on the Amplitude and Phase; Plus Phase Jitter. The noise and jitter conditions were simulated within the DATINT subroutine by using data values:

SIGMA =	3.0
PJPER =	3.0
PJAMP =	10.0
TARGET =	1.0
NPD =	1.0
AH =	0.0
PH =	0.0

These values cause Equations 1 and 2 to provide amplitude and phase values with noise as in the previous example, but Equation 2 also adds a cosine jitter component with magnitude 10 and period of 3 received points or 800 Hz at 2400 baud.

The computer signal space pattern display in Figure 14A hardly shows the presence of phase jitter and even the amplitude and phase deviations display in Figure 14B fails to obviously show phase jitter (though it does show some consistent deviations at each extreme). However, the phase deviation frequency spectrum in Figure 14C shows a large frequency magnitude at 800 Hz which would be faster than the human eye could even detect on the oscilloscope display of the signal space pattern.

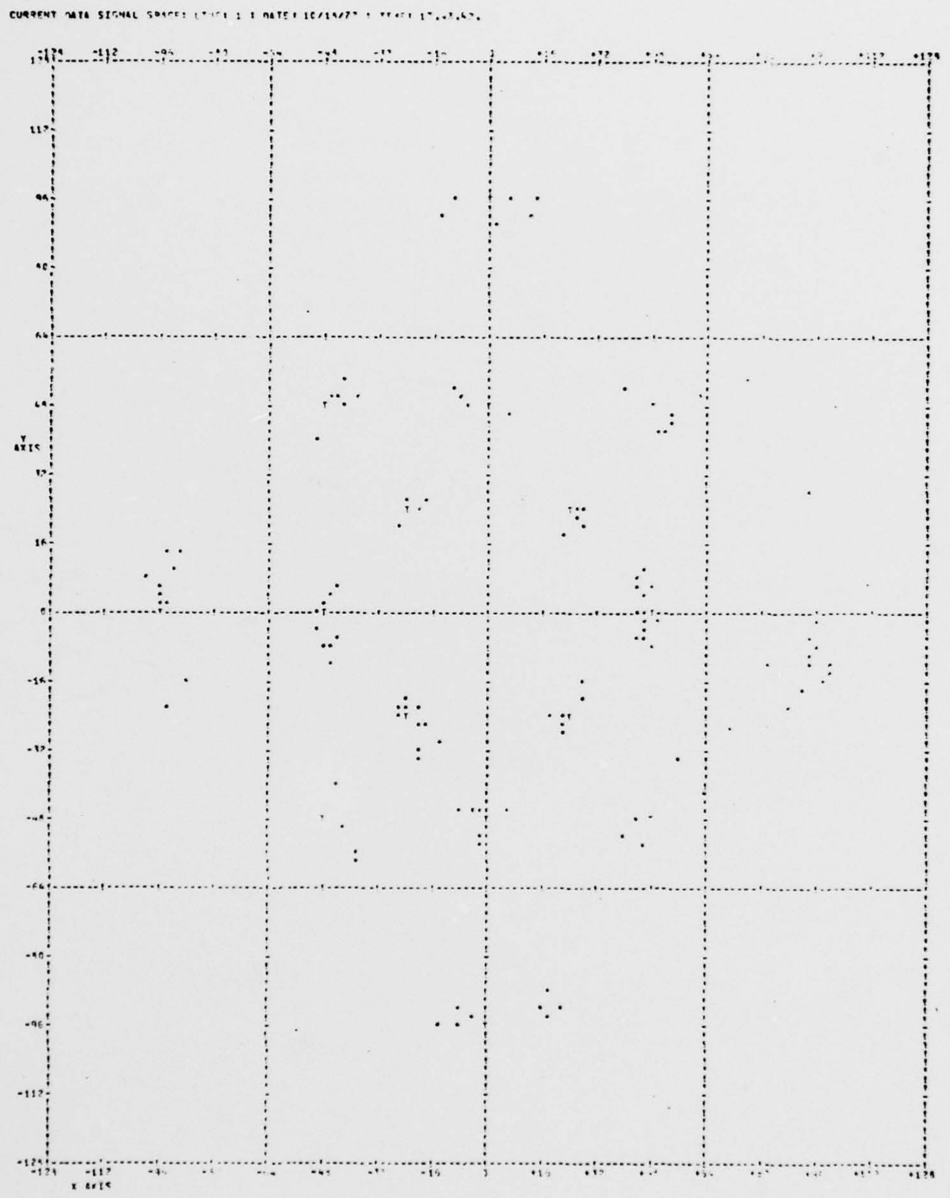


Figure 14A. Current Data Display: Normal Gaussian Noise on the Amplitude and Phase with Phase Jitter Added (Signal Space Pattern)

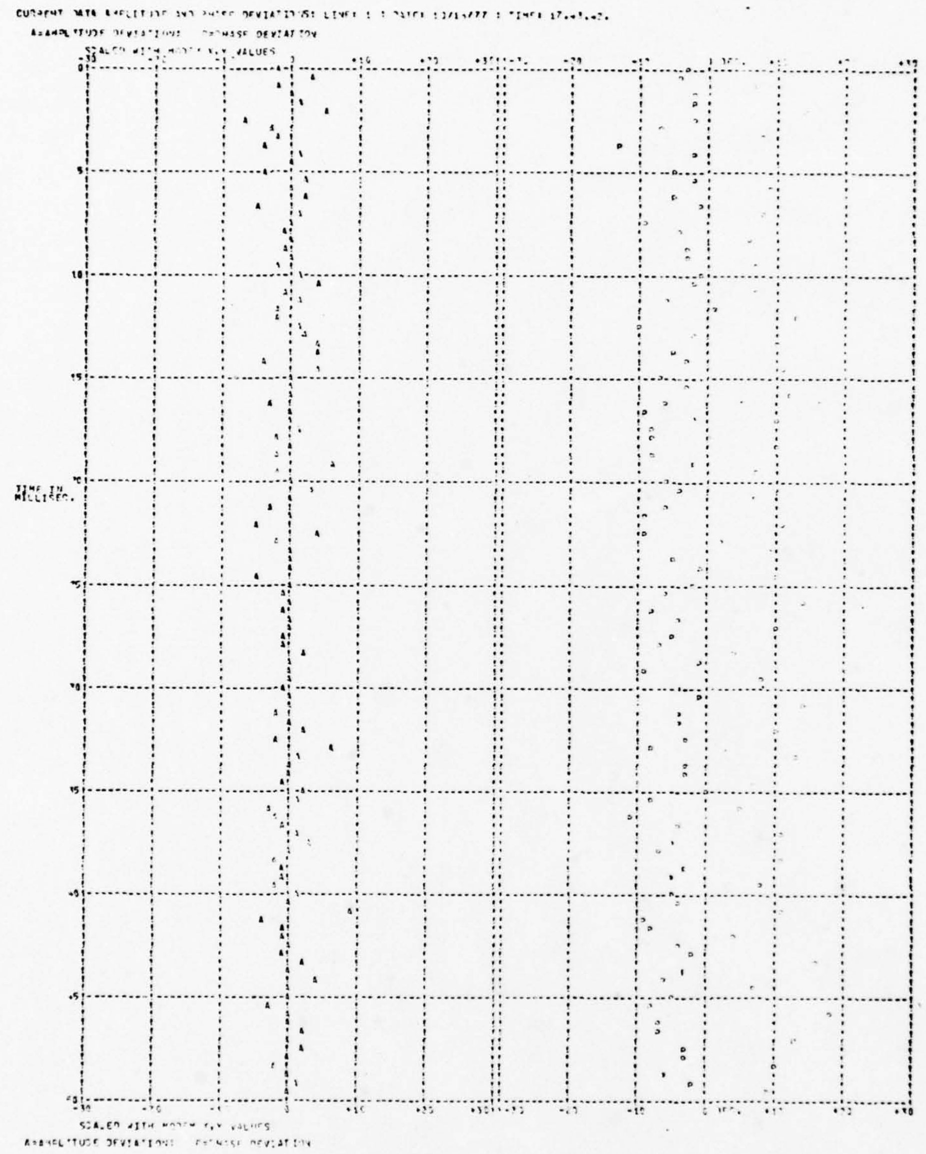


Figure 14B. Current Data Display: Normal Gaussian Noise on the Amplitude and Phase with Phase Jitter Added (Amplitude and Phase Deviations)

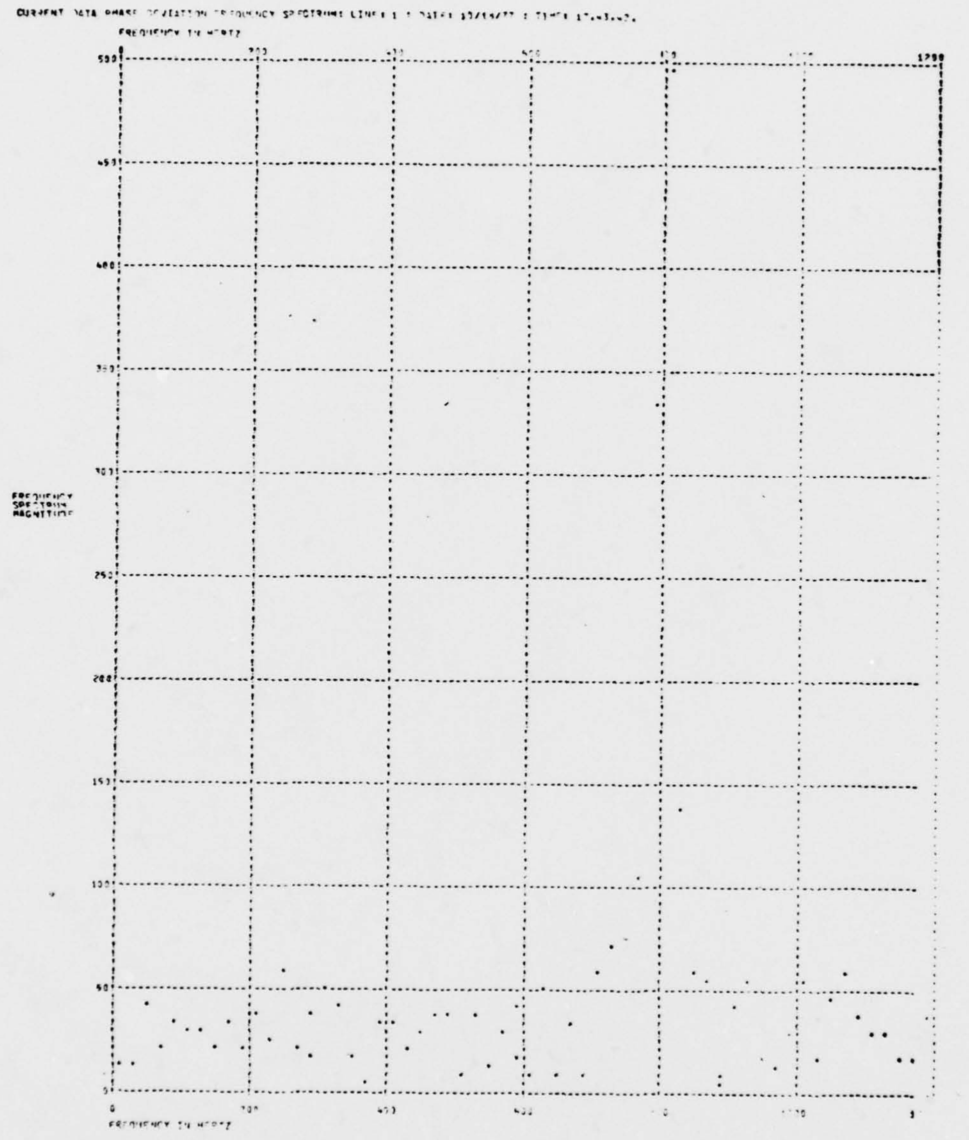


Figure 14C. Current Data Display: Normal Gaussian Noise on the Amplitude and Phase with Phase Jitter Added (Phase Deviation Frequency Spectrum)

Phase and Amplitude Hits. The phase and amplitude hits were simulated within the DATINT subroutine using data values:

		Amplitude Hit	Phase Hit
SIGMA	=	3.0	3.0
PJPER	=	1.0	1.0
PJAMP	=	0.0	0.0
TARGET	=	1.0	1.0
NPD	=	1.0	1.0
AH	=	10.0	0.0
PH	=	0.0	10.0

These values caused Equations 1 and 2 to provide noisy values for amplitude and phase, and they also added a constant deviation to the amplitude or phase to simulate the hit.

The computer signal space pattern display in Figure 15A and Figure 16A respectively show amplitude and phase hits as in the oscilloscope signal space patterns in Figure 4.

The amplitude and phase deviation displays in Figures 15B and 16B show the quantitative values of the hit deviations. They also provide a rough indication of the mean deviation (10.0 in both sample displays of amplitude and phase hits).

The phase deviation frequency spectrum simply shows that the phase deviations are varying randomly.

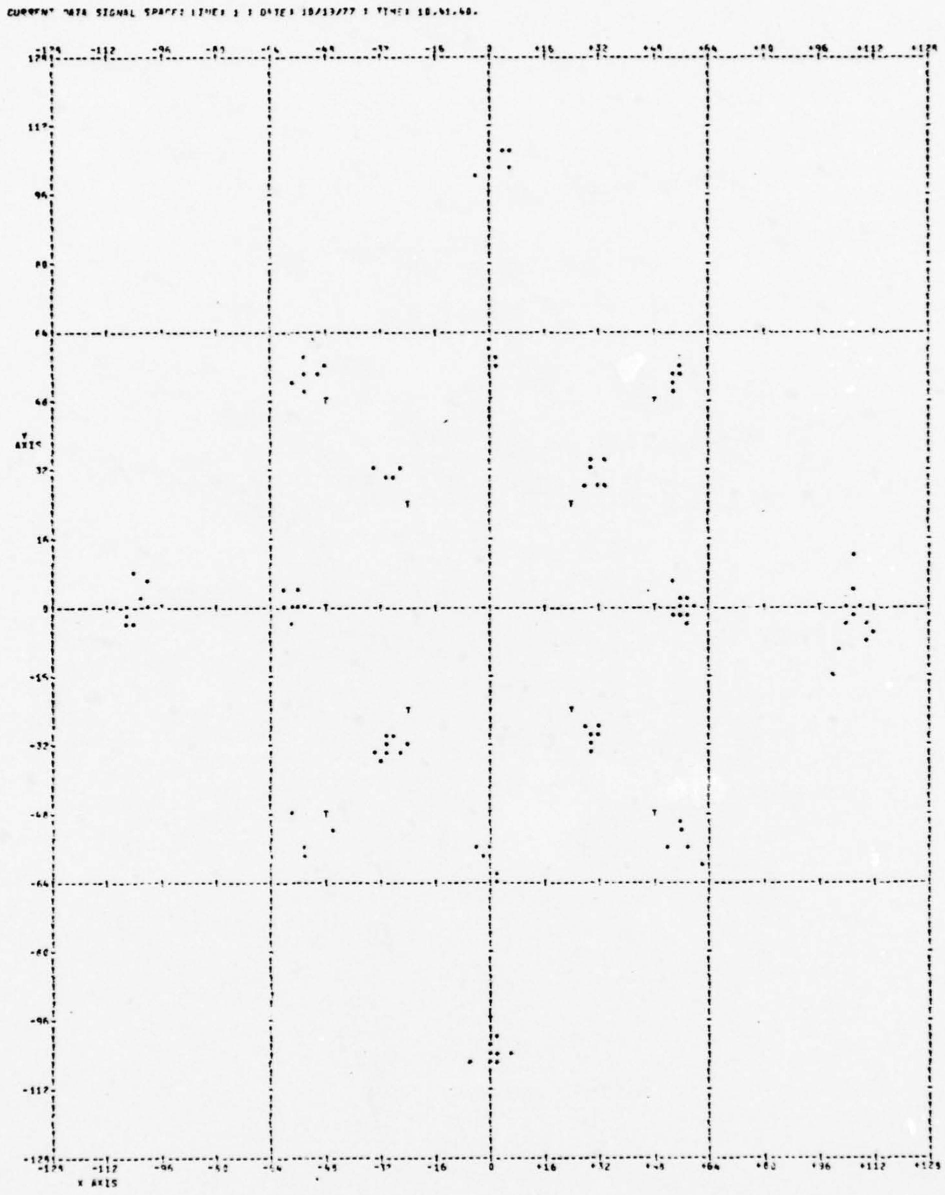


Figure 15A. Current Data Displays: Normal Gaussian Noise Plus an Amplitude Hit (Signal Space Pattern)

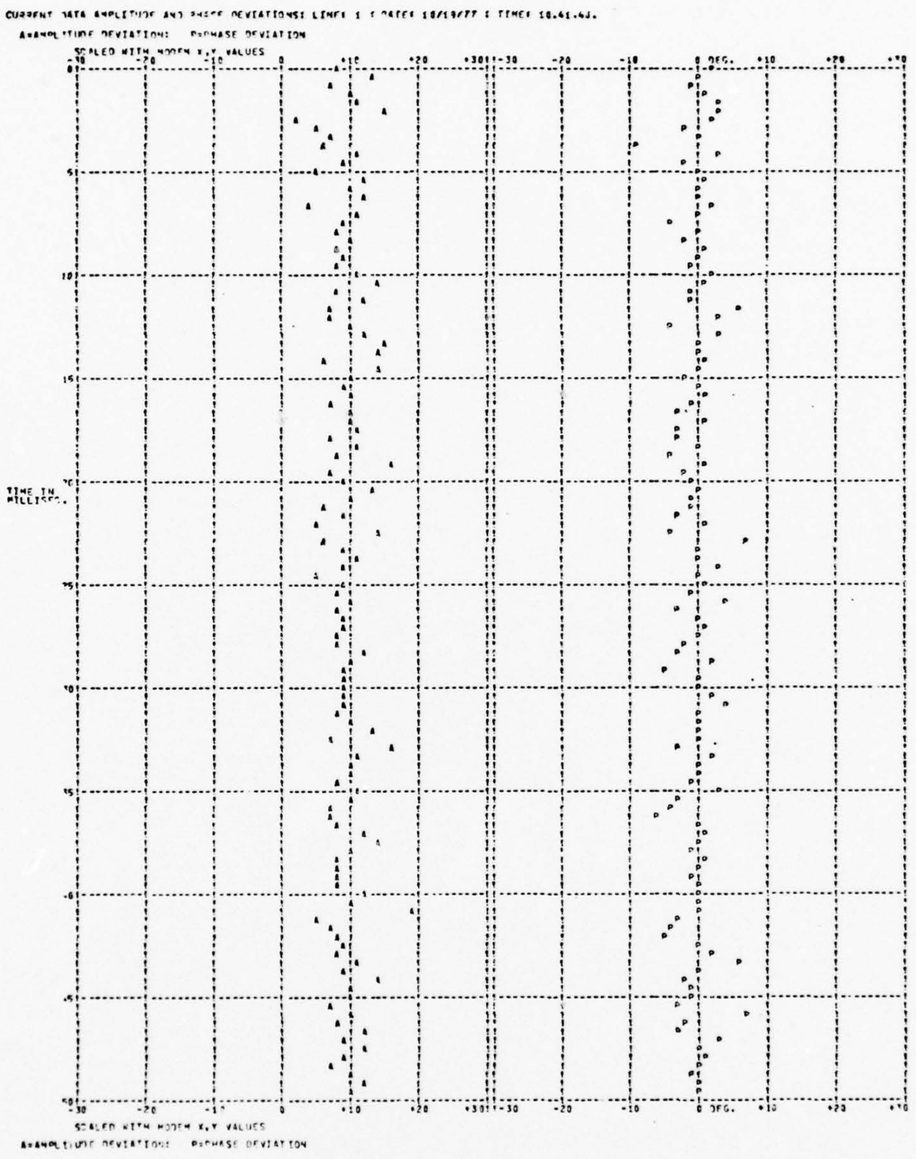


Figure 15B. Current Data Display: Normal Gaussian Noise Plus an Amplitude Hit (Amplitude and Phase Deviations)

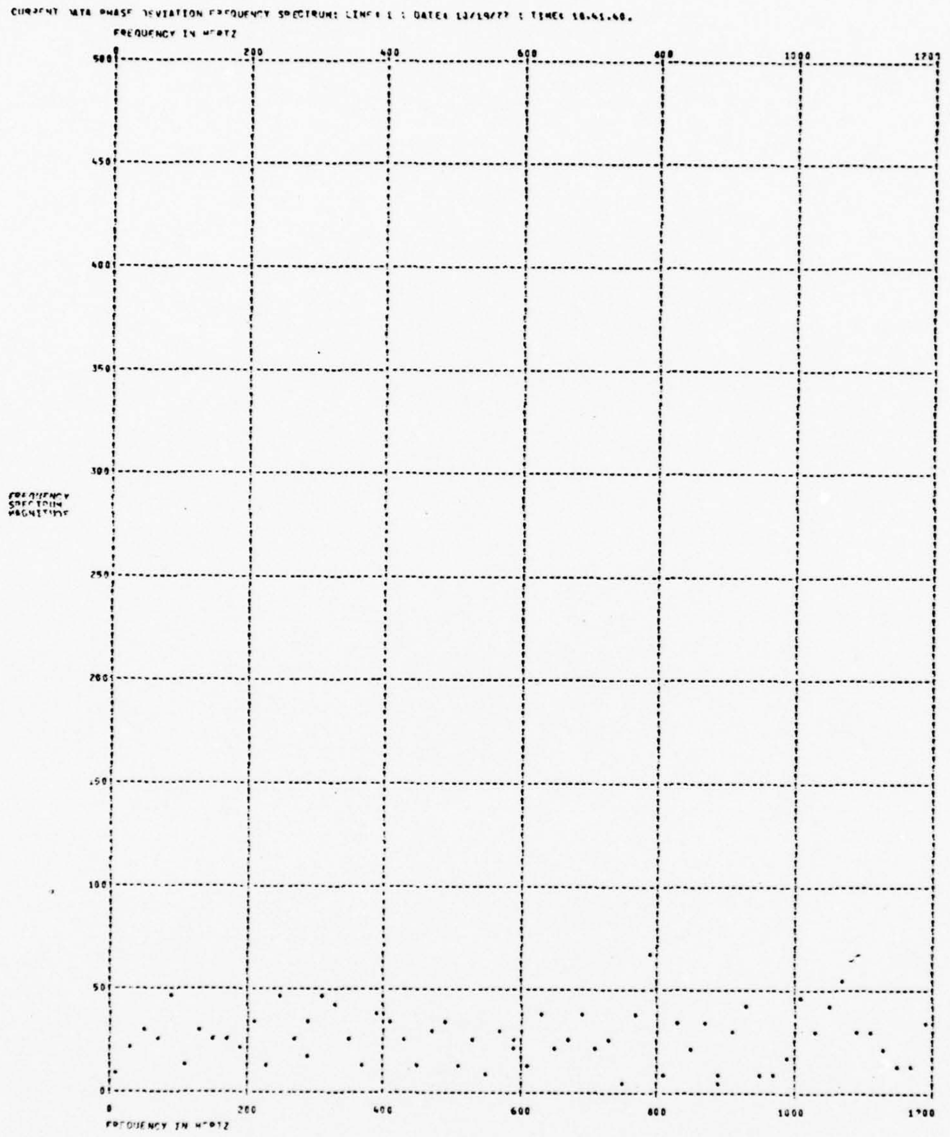


Figure 15C. Current Data Display: Normal Gaussian Noise Plus an Amplitude Hit (Phase Deviation Frequency Spectrum)

CURRENT DATA SIGNAL SPACE: LEVEL 1 : DATE: 10/13/77 : TIME: 16.13.46.

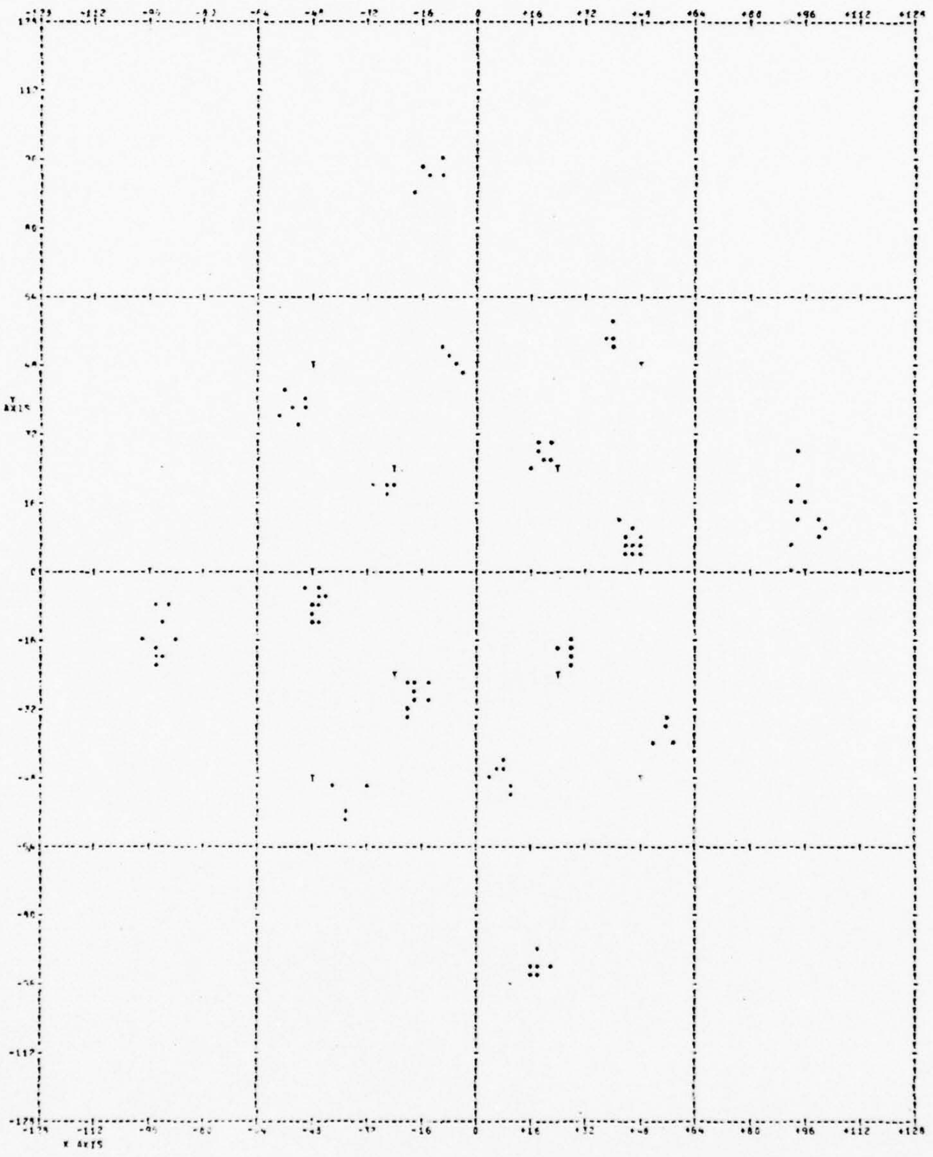


Figure 16A. Current Data Display: Normal Gaussian Noise Plus a Phase Hit (Signal Space Pattern)

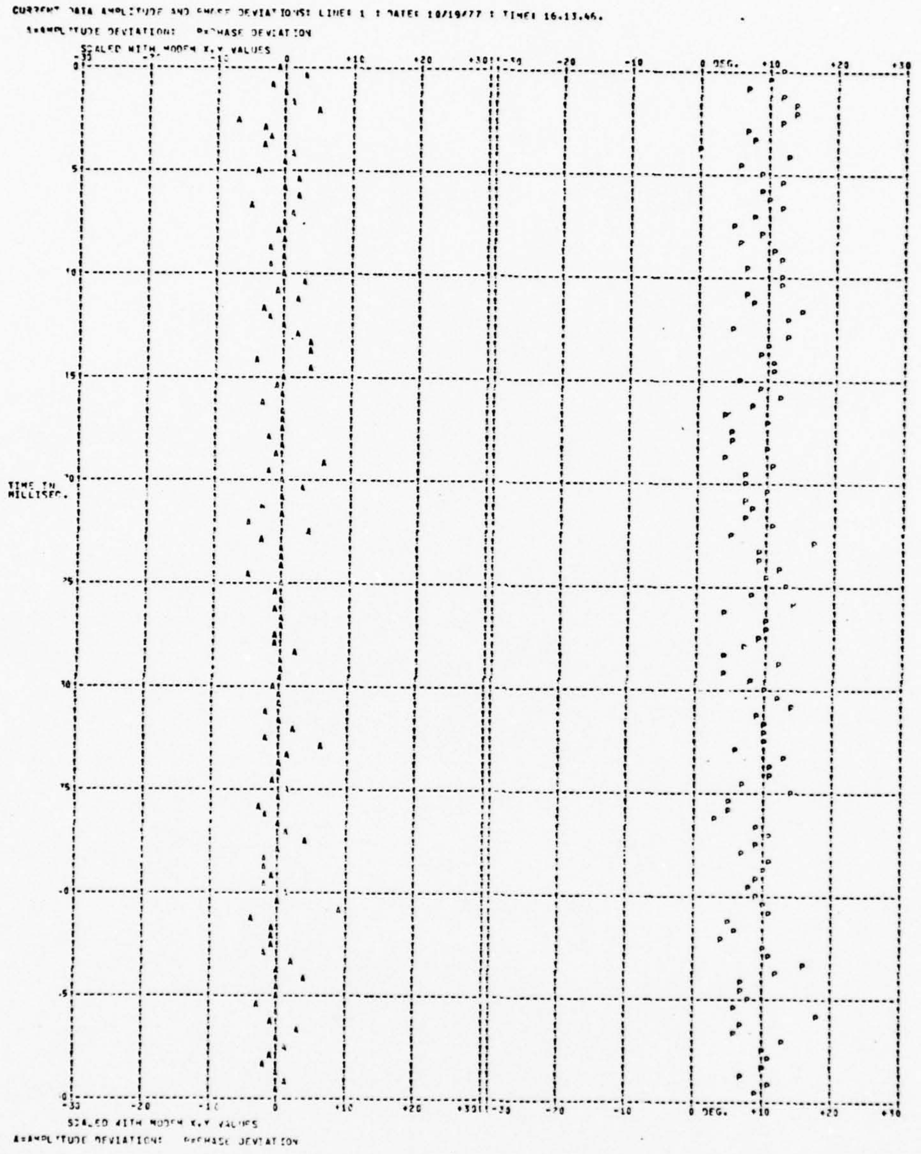


Figure 16B. Current Data Display: Normal Gaussian Noise Plus a Phase Hit (Amplitude and Phase Deviations)

CURRENT DATA PHASE DEVIATION FREQUENCY SPECTRUM: LINE: 1 : DATE: 10/19/77 : TIME: 15.13.46.

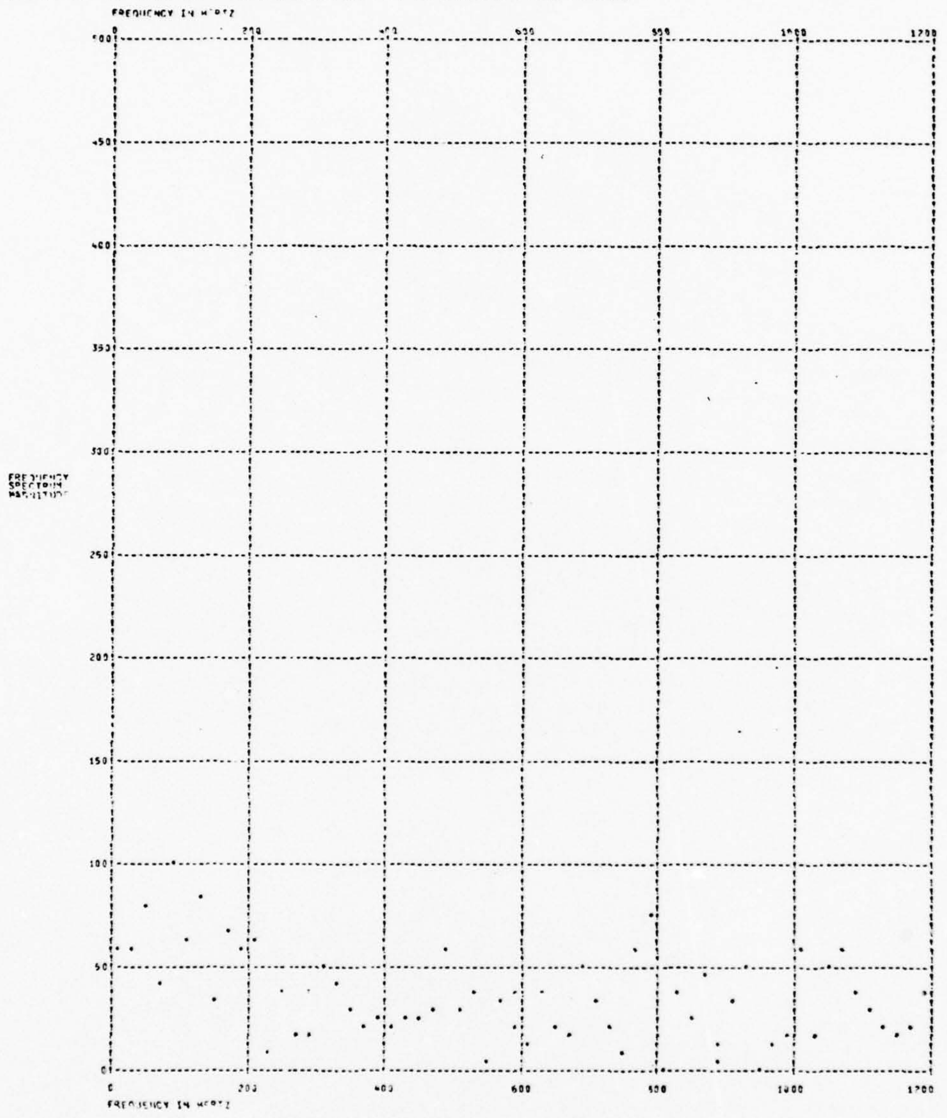


Figure 16C. Current Data Display: Normal Gaussian Noise Plus a Phase Hit (Phase Deviation Frequency Spectrum)

16C

Limitation of Current Data Display

Though the current data displays offer more qualitative information than just the oscilloscope display of the signal space pattern, there are still several limitations and impracticalities in the use of the current data displays. To determine noise levels and relative transmission line qualities, comparison sets of displays with known values of perturbations are required. Also, the amount of paper and time utilized in printing the displays prevents them from being utilized constantly as a monitoring system, and consequently line perturbations which only happen sporadically will be difficult to display. Therefore, the current data displays are not sufficient for detection of perturbation trends or for detection of occasional problems. The current displays are best suited for observing transmission line perturbations when a problem is already known to exist on the line and it needs to be identified and measured.

V. Recommendations

Though the current data analysis system developed in this thesis has some limitations, it would provide real time quantitative information about transmission line perturbations that could not be obtained from other available monitor systems. Therefore it would be worthwhile to implement this system, especially since all of the necessary equipment would already be available within the RADC laboratories. Also, it would be easier to initially derive the minicomputer program from the FORTRAN program and develop the keyboard controlling routines with just the current data modules. Although the program structure was developed with an expansion capability included in the basic design, several considerations that were not critical in the current data system would have to be considered in programming the expanded system. If the current data system was implemented first, then the extra considerations for the expanded system could be defined more clearly at the beginning of the expansion design and development.

The design and development of an expanded system would first require determination of all the operations which would be required to convert the data from the current data modules to appropriate values for a permanent history. Then the program structure to accomplish those operations would have to be developed along with considering all of the additional timing constraints which would arise from the additional processing operations. Just the determination of the operations required to provide values for means and variances of amplitude deviation, phase deviation, phase dispersion, noise, line quality and to provide the number of severe or medium phase or amplitude hits would be a complex, time consuming task. Then, to figure out how to get it all done in only 49,920 microseconds (the time frame between input buffers of 120 points each) would be another monumental piece of work. Also, the cataloging, filing, and retrieving routines, plus the display routines would have to be worked out for the different summaries. The IBM-LQM system was developed with the expanded capabilities, but it was developed by a team of engineers, statisticians, and programmers over a seven month period (Ref. 1). Though such a system could conceivably require several thesis efforts, the results would be well worthwhile.

Bibliography

1. Bryant, P., F.W. Giesin, and R.M. Hayes. "Experiments in Line Quality Monitoring". IBM Systems Journal, 15: (#2, 1976).

2. Codex. LSI 9600 High Speed Data Modem. Operation Manual. Newton, Massachusetts: Codex Company, June 1976.

3. Marvin, K.L. Capt. Structured Analysis and Design of a Satellite Simulator Thesis. GE/EE/77S-6. Wright Patterson Air Force Base, Ohio: Air Force Institue of Technology, Engineering, September 1977.

4. Oppenheim, Alan V., and Ronald W. Schafer. Digital Signal Processing. New Jersey: Prentice Hall, INC., 1975.

5. Digital Equipment Corporation. PDP-11/04/05/10/35/40/45 Processor Handbook. Maynard, Massachusetts: Digital Equipment Corporation, 1975.

6. Digital Equipment Corporation. PDP-11 Peripherals Handbook. Maynard, Massachusetts: Digital Equipment Corporation, 1976.

APPENDIX A

Structured Program Development

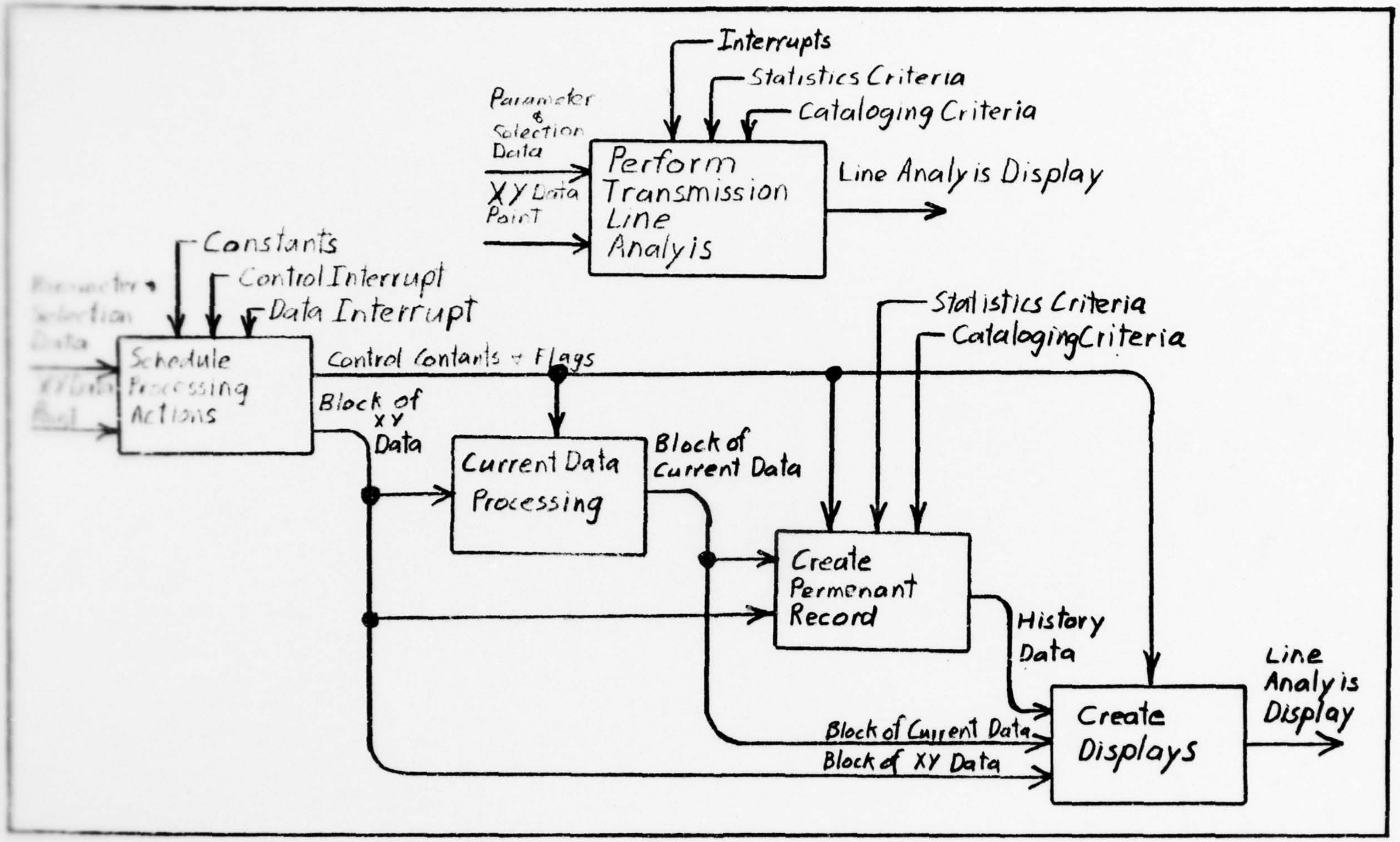


Figure A-1. First Two Levels of Structured Program

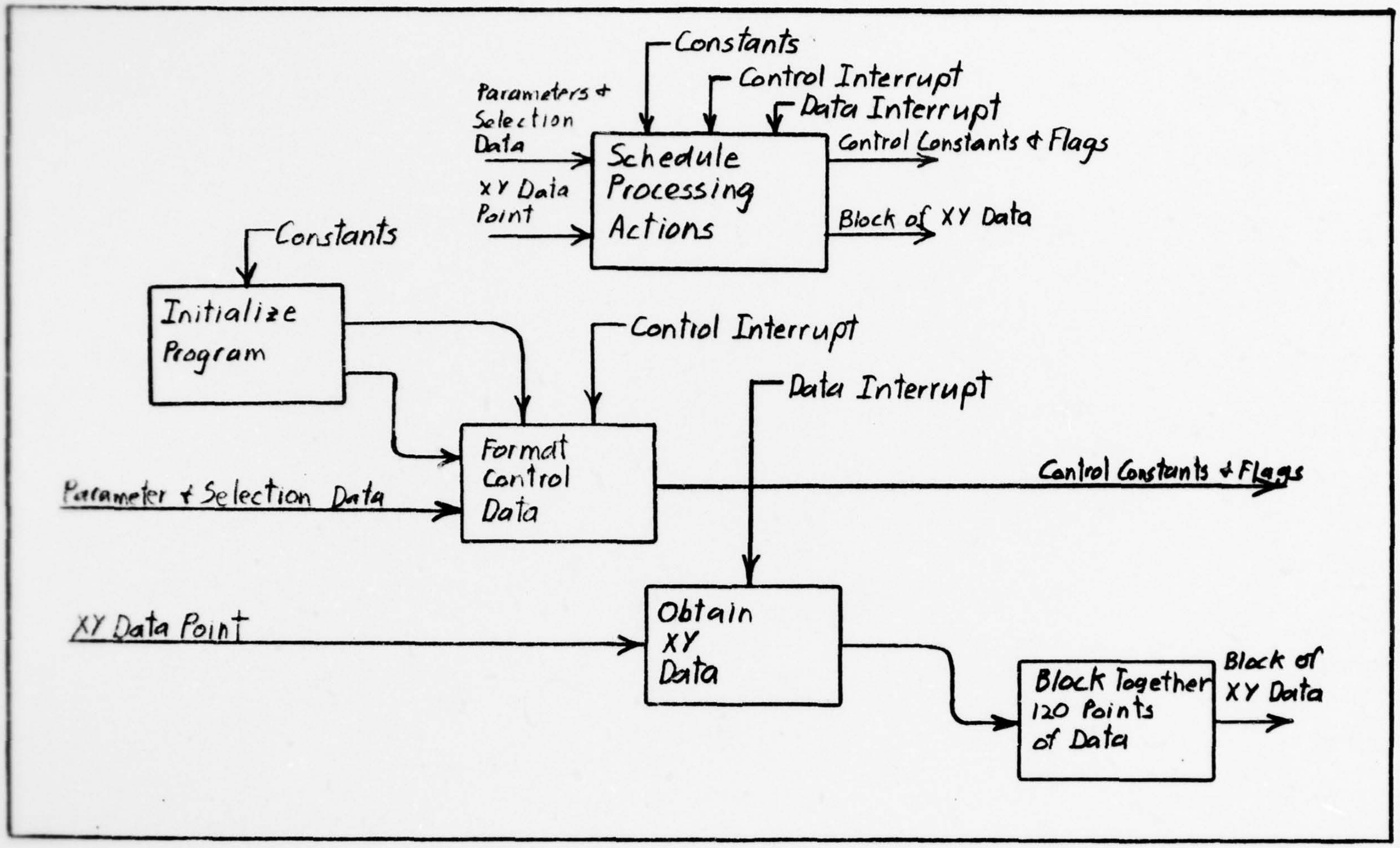


Figure A-2. Third Level Efferent Block

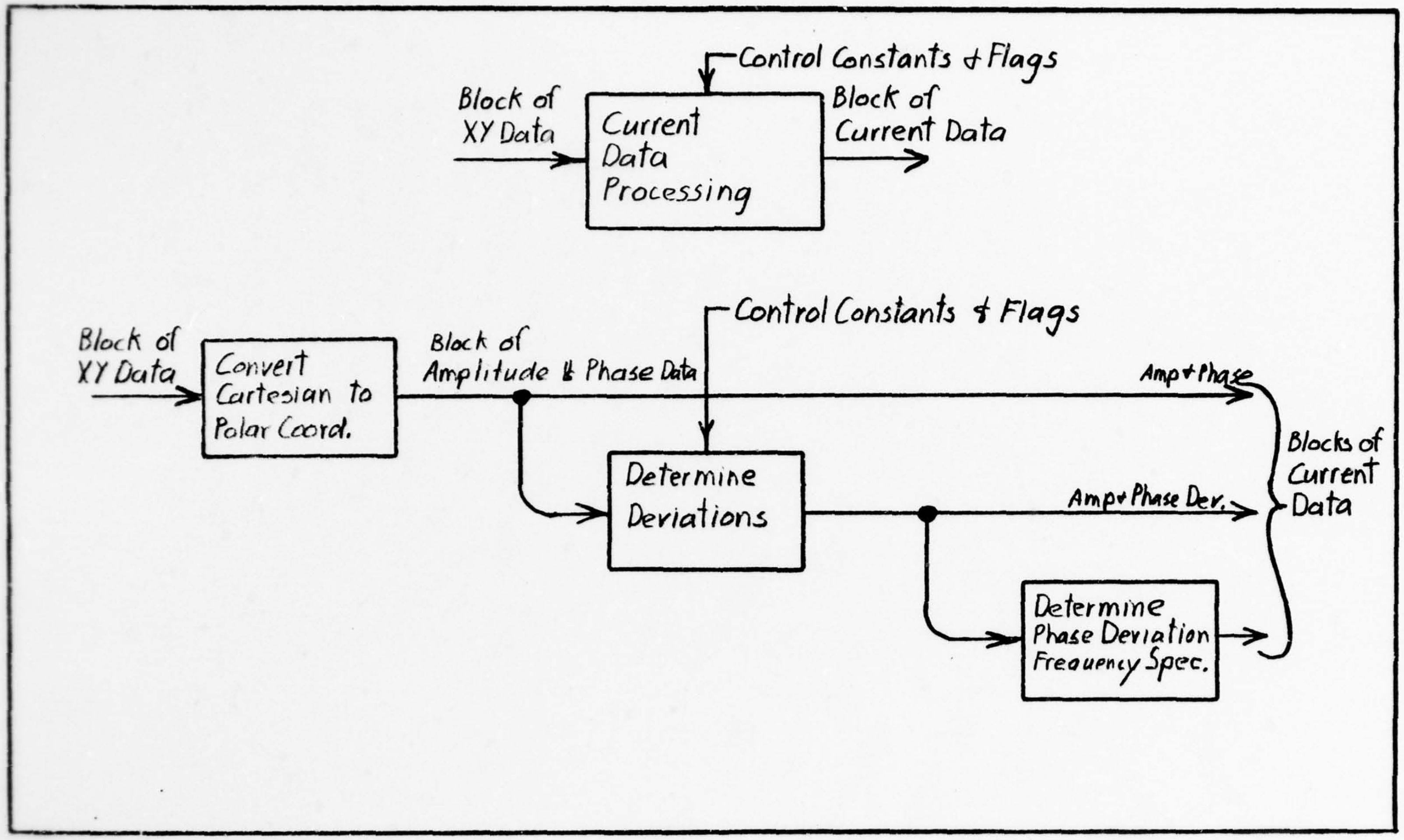

Control Constants & Flags
Block of XY Data
Current Data Processing
Block of Current Data
Control Constants & Flags
Block of XY Data
Convert Cartesian to Polar Coord.
Block of Amplitude & Phase Data
Amp+Phase
Determine Deviations
Amp+Phase Dev.
Blocks of Current Data
Determine Phase Deviation Frequency Spec.

Figure A-3. Third Level Transitional Block (Current Processing)

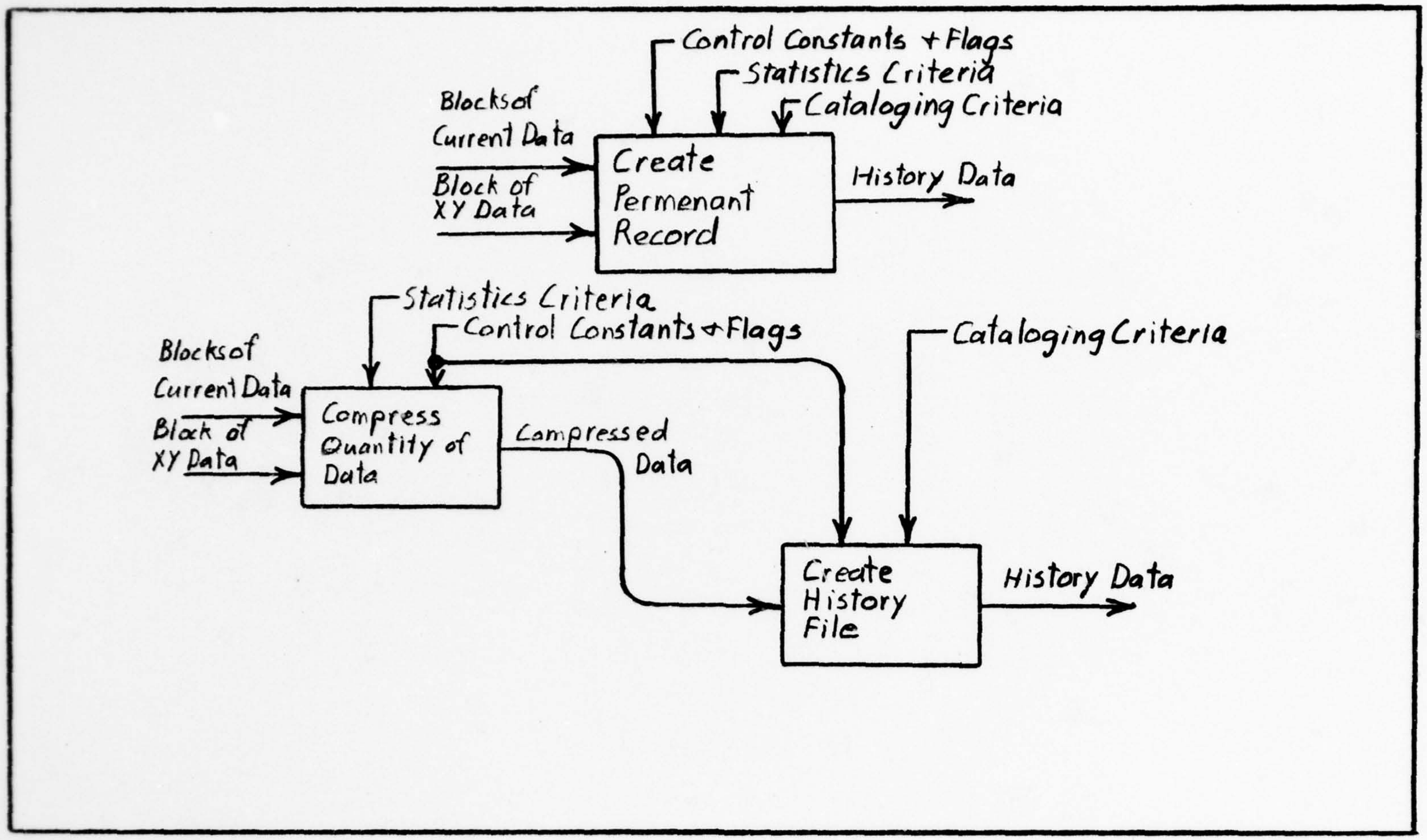


Figure A-4. Third Level Transitional Block (Permanent Record)

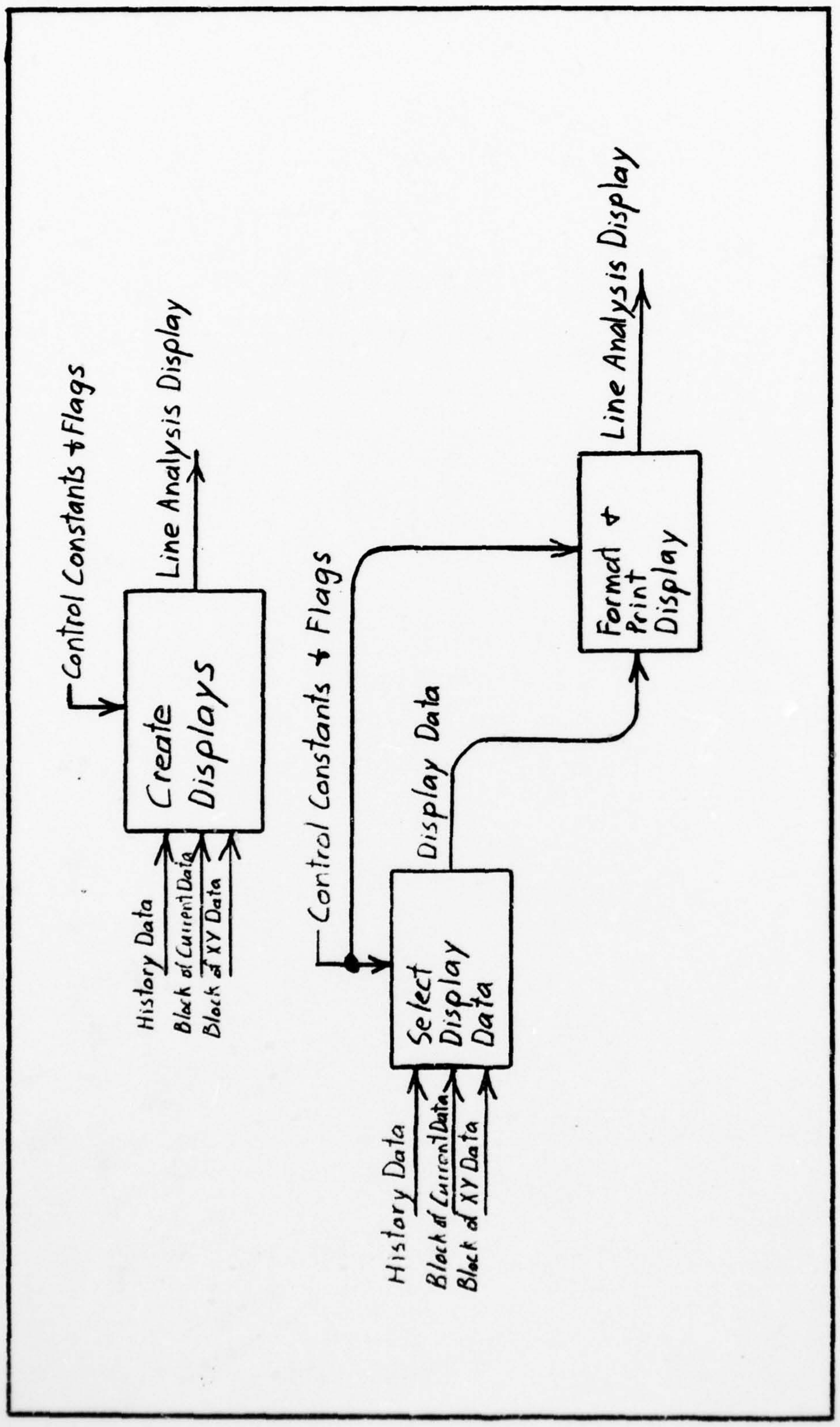


Figure A-5. Third Level Afferent Block

APPENDIX B

FORTRAN Program Flow Diagrams

(Calling Routines Only)

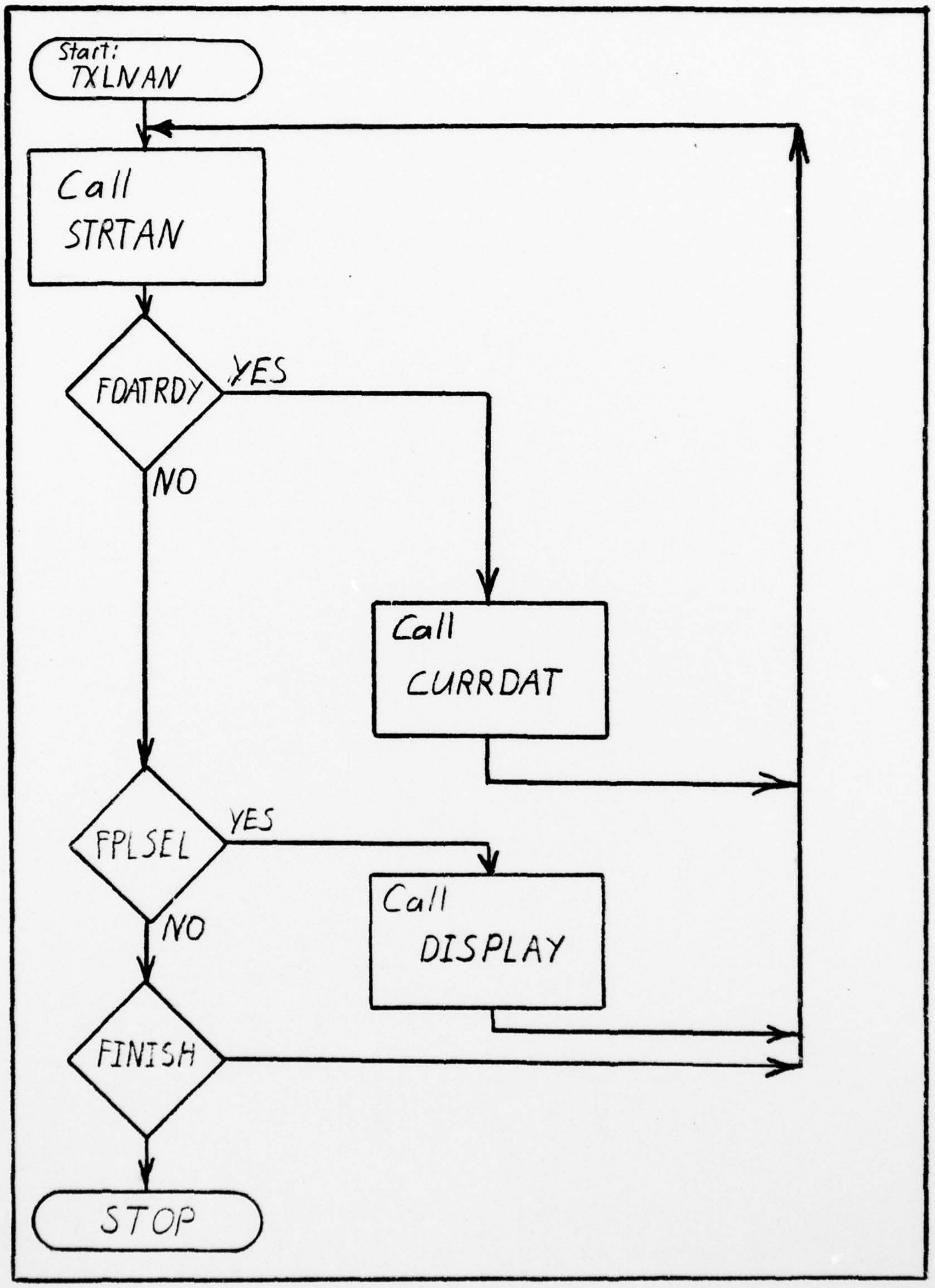


Figure B-1. Program TXLNAN

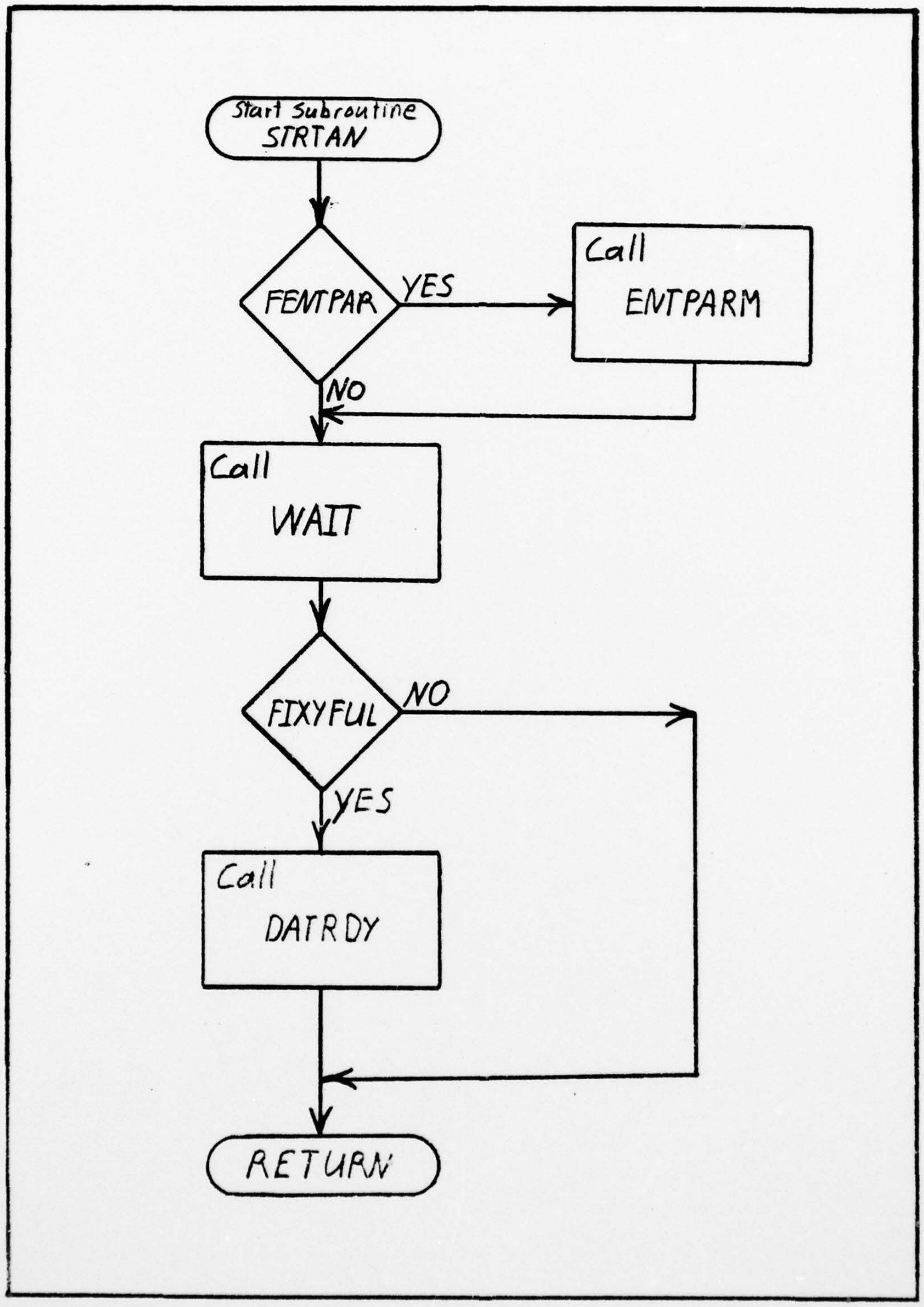


Figure B-2. Subroutine: STRTAN

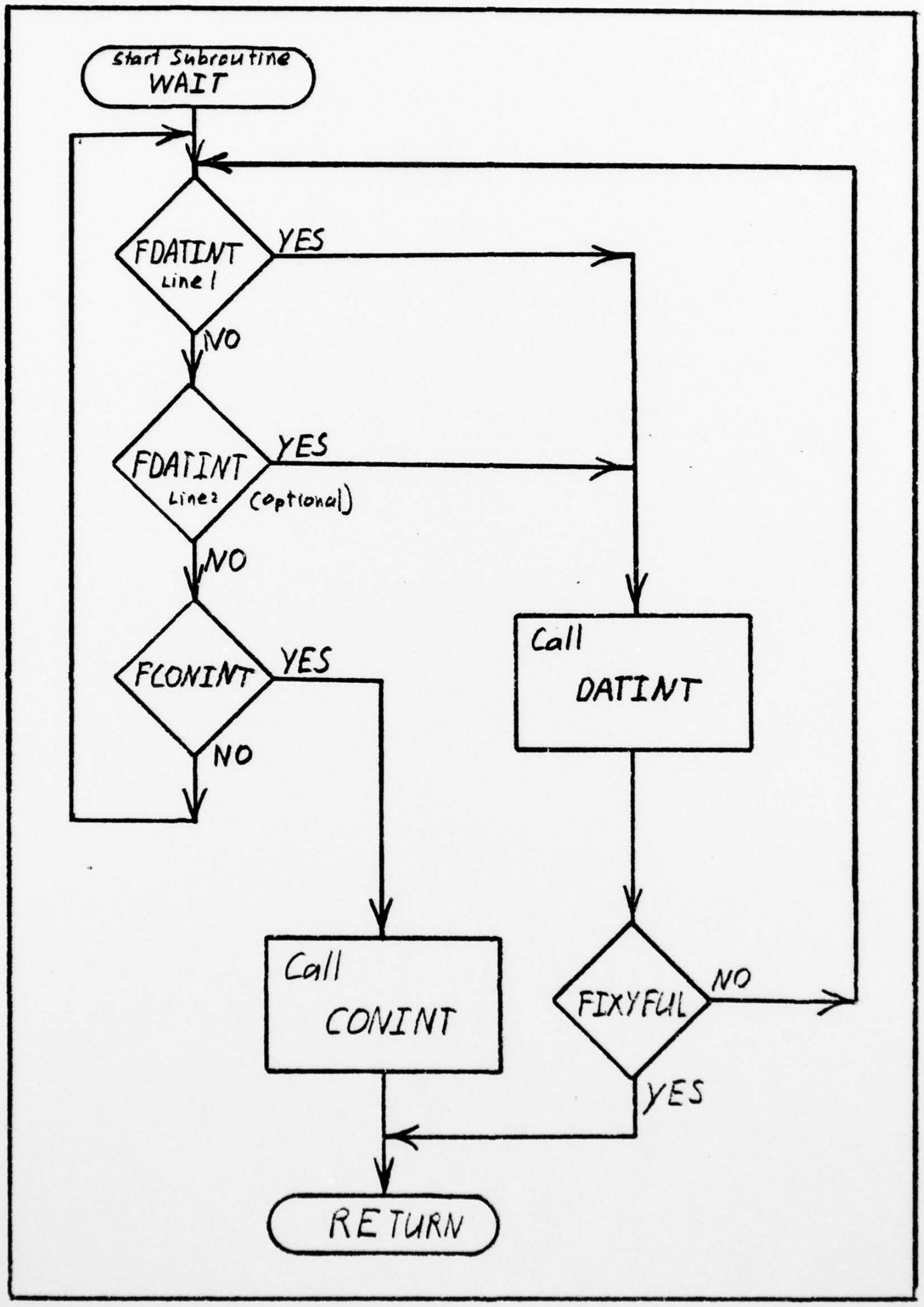


Figure B-3. Subroutine: WAIT

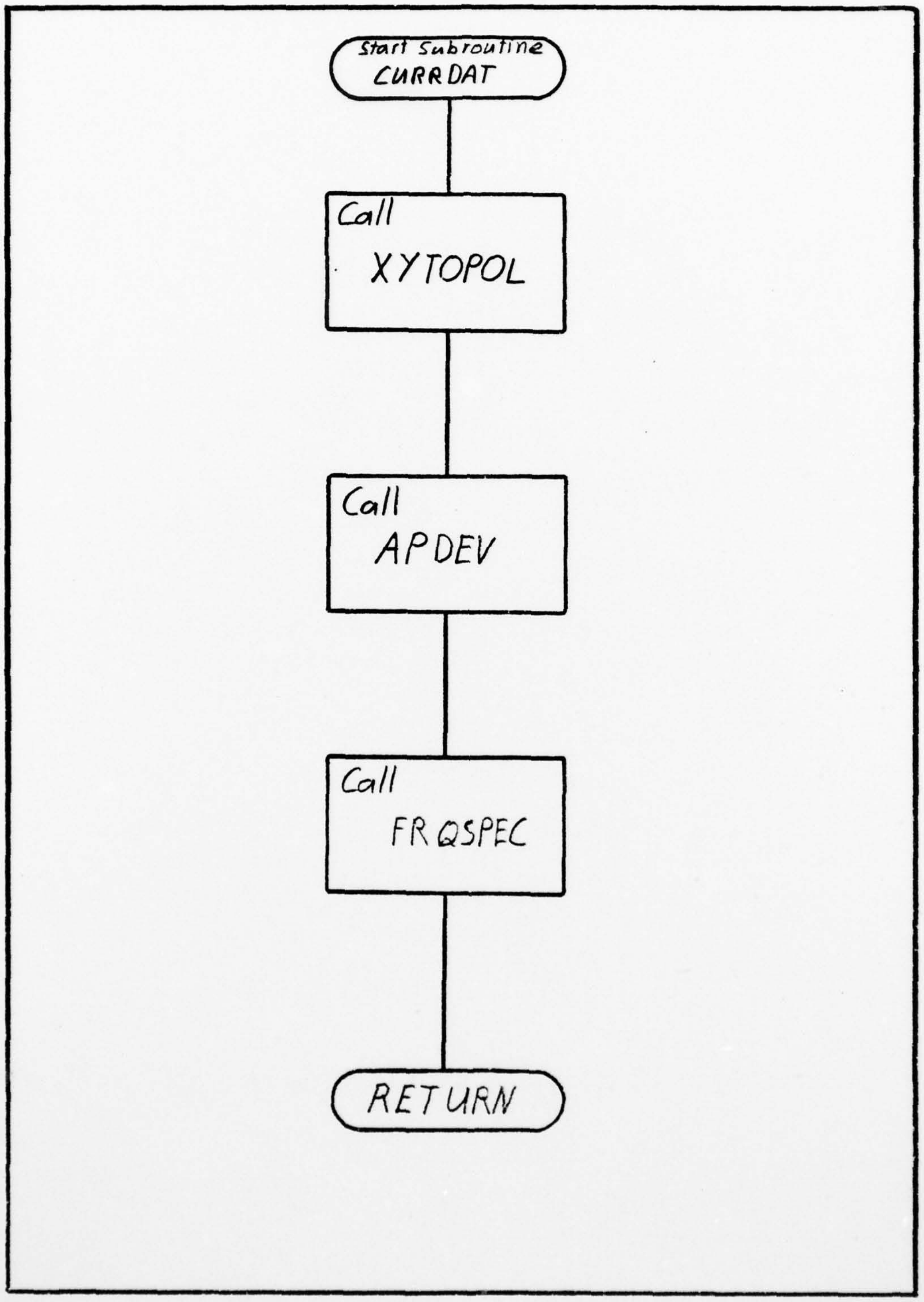


Figure B-4. Subroutine: CURRDAT

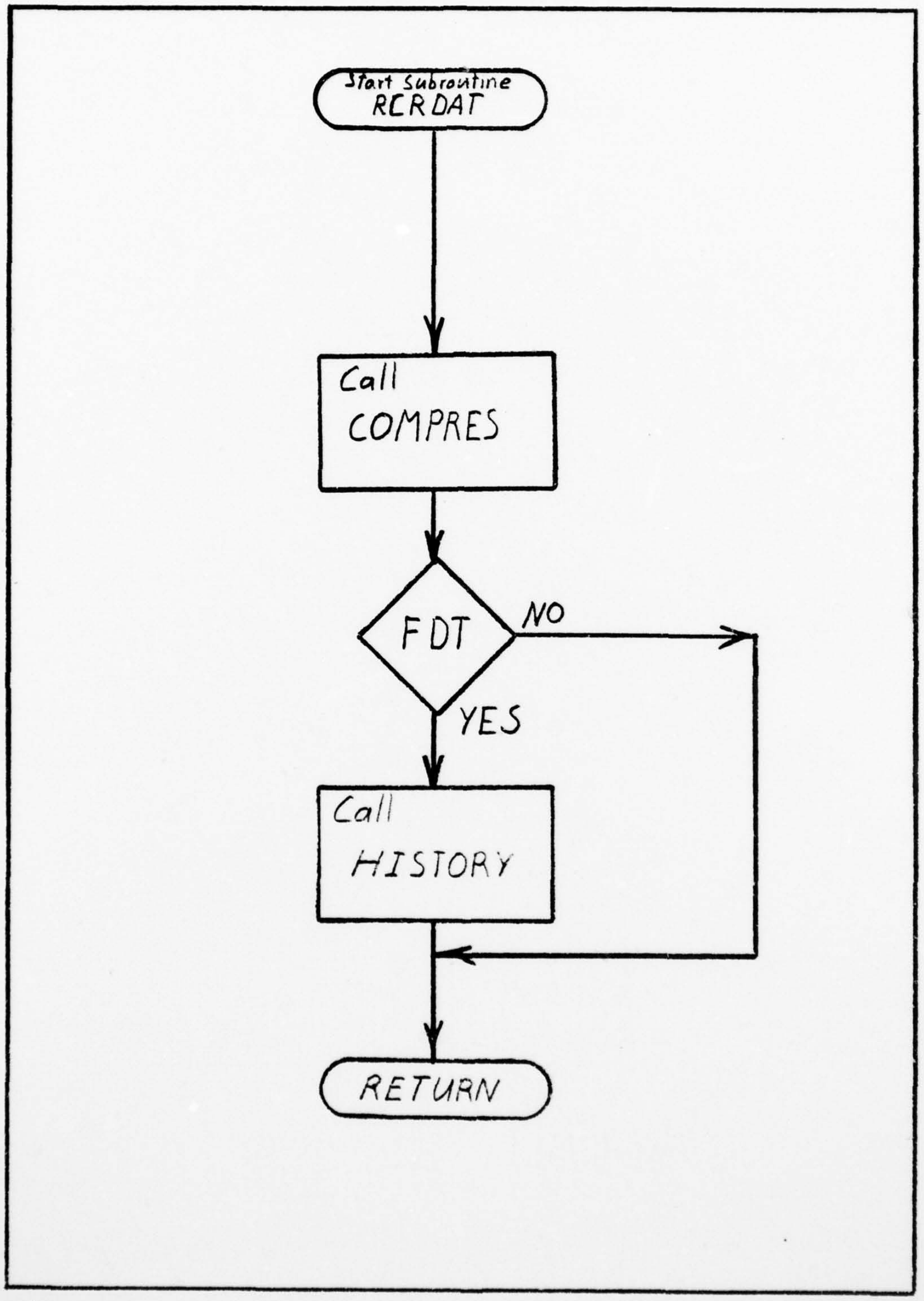


Figure B-5. Subroutine: RCRDAT

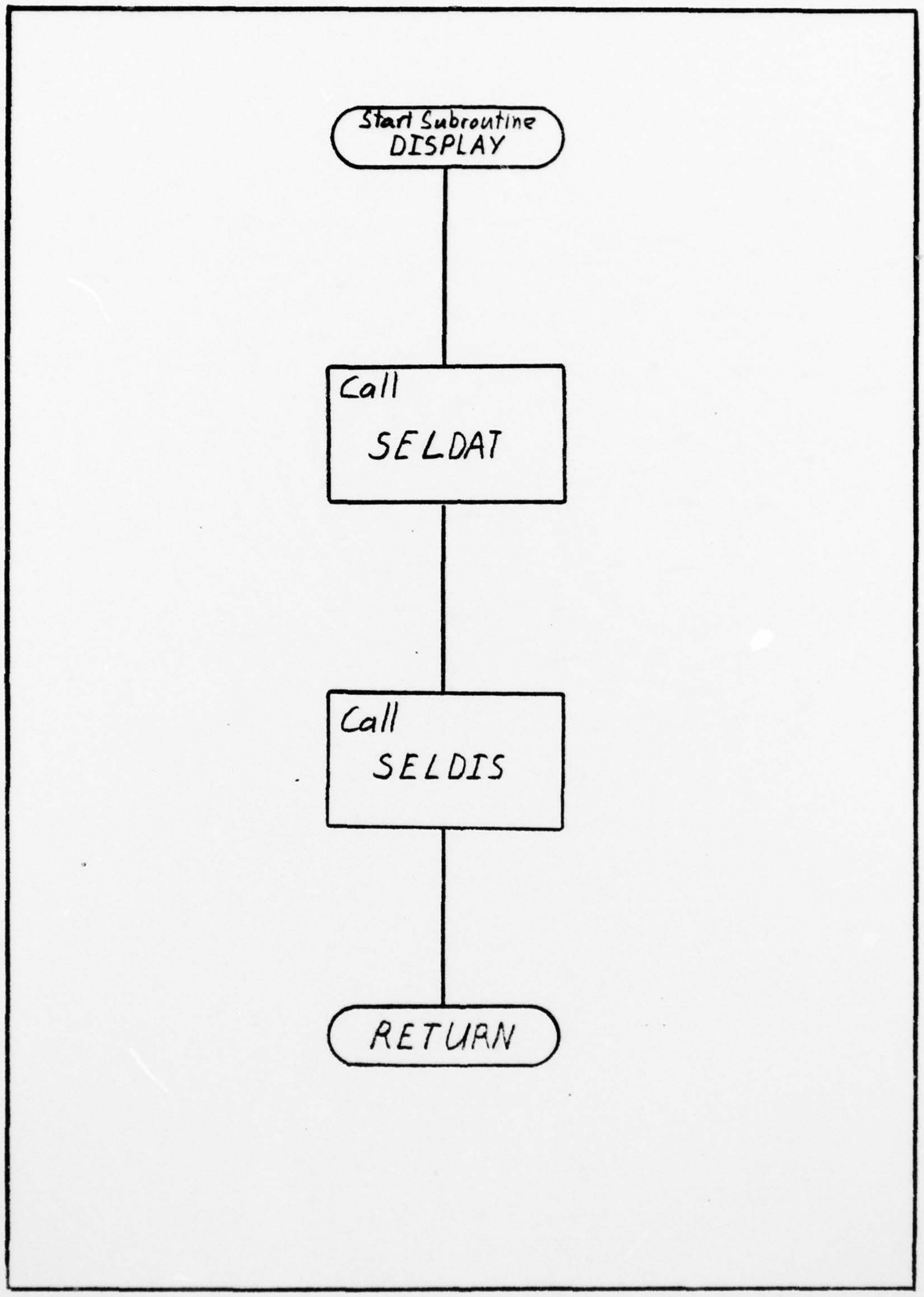


Figure B-6. Subroutine: DISPLAY

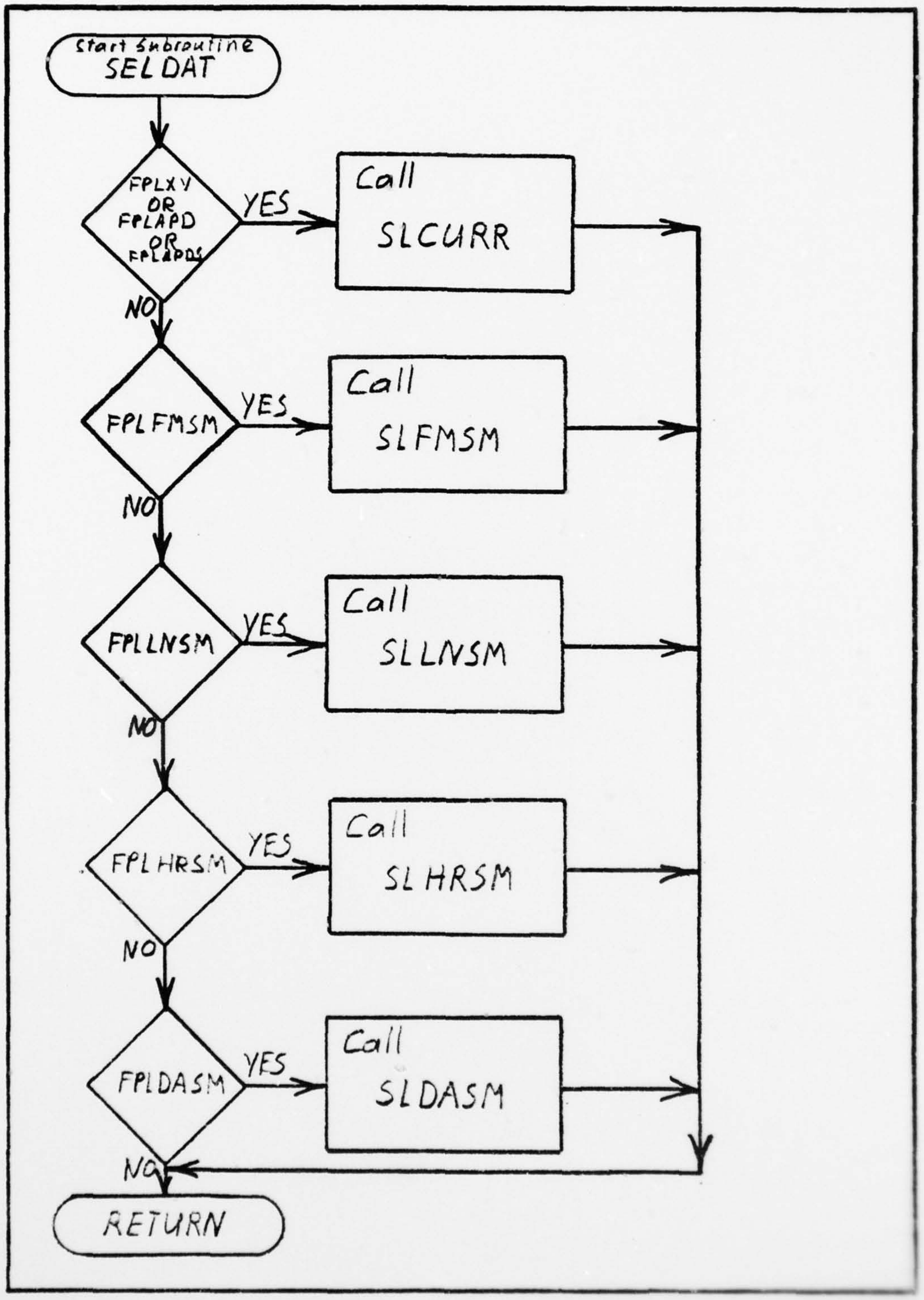
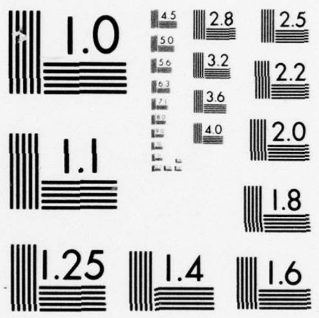


Figure B-7. Subroutine: SELDAT


1.0
4.5
5.0
5.6
6.3
7.1
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

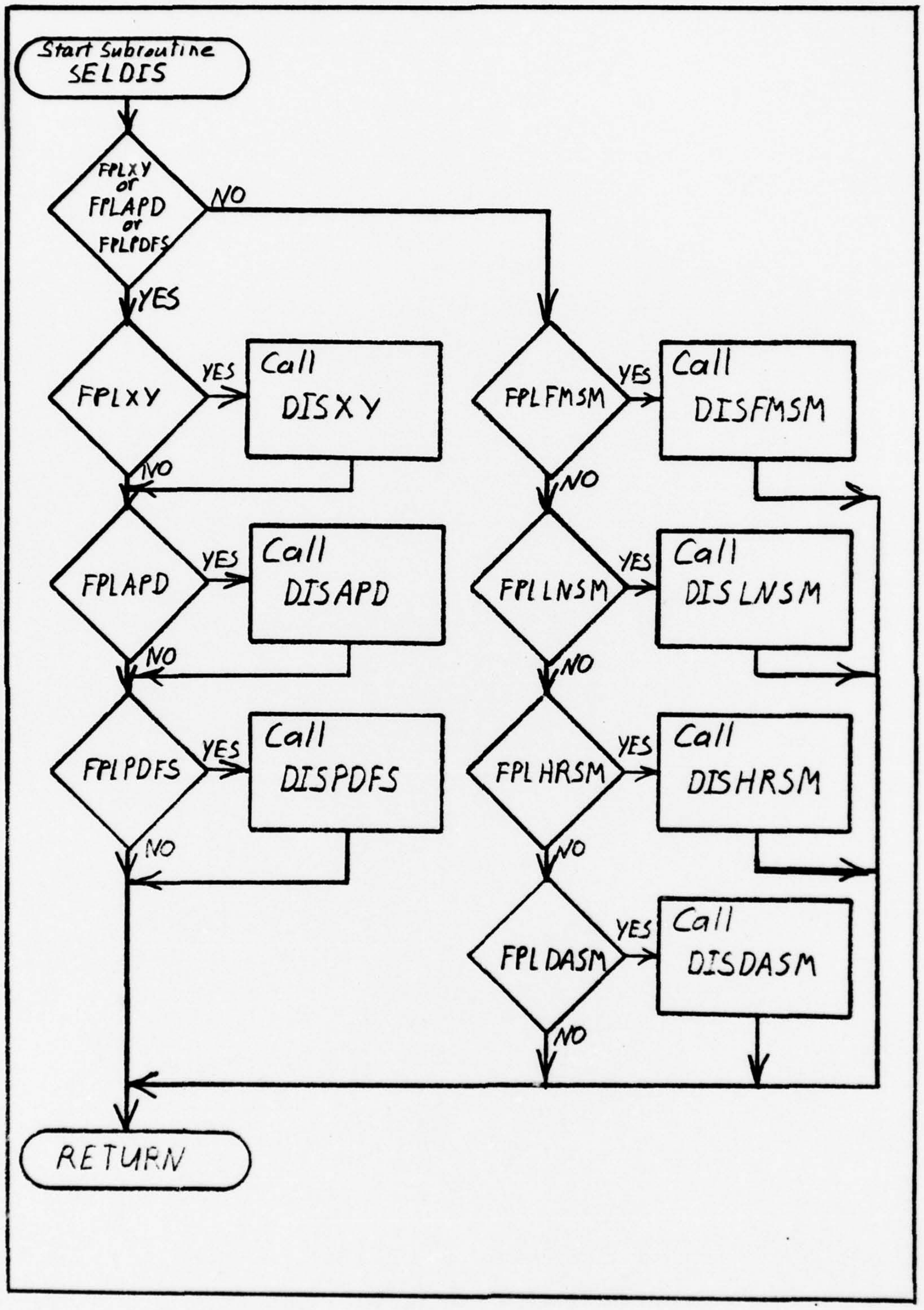


Figure B-8. Subroutine: SELDIS

APPENDIX C

FORTRAN Program

```
      PROGRAM TXLNAN (INPUT,OUTPUT)

C MASTER CONTROL PROGRAM FOR TRANSMISSION LINE ANALYSIS

C     CURRENT DATA STORAGE AND PLOT FILES
        COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
        COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
        COMPLEX PFPDFS(120)
C     LINE # BEING PROCESSED
        INTEGER LN,LNSEL,YES,NO,LNC
C     SUBROUTINE DECISION FLAGS
        INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
        INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
        INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPL4RSM,FPLDASM,FPLAPD
C     TIME PARAMETERS
        REAL DT,LNTIME(2)

        COMMON   XYT,IXY,XY,PFXYT,PFXY
        COMMON   APT,AP,APD,PDFS,PFAPD
        COMMON   PFPDFS
        COMMON   LN,LNSEL,YES,NO,LNC
        COMMON   FCONINT,FDATINT,FLO,FLI,FDT
        COMMON   FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
        COMMON   FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPL4RSM,FPLDASM,FPLAPD
        COMMON   DT,LNTIME,DAYT,TYME
        COMMON   PLOTMAP(130,130)

C     SET ENTER PARAMETERS FLAG
        FENTPAR=YES

C     INITIATE AND/OR WAIT FOR INTERRUPTS AND FULL INPUT BUFFERS
1       CALL STRTAN

C     SELECT STARTAN, CURRDAT, DISPLAY, OR STOP
        IF(FDATRDY.EQ.YES) GO TO 2
        IF(FPLSEL.EQ.YES) GO TO 3
        IF(FINISH.EQ.YES) GO TO 4
        GO TO 1

C     PROCESS CURRENT DATA
2       CALL CURRDAT
        CALL STORDAT
        GO TO 1

C     CREATE DISPLAY OUTPUT
3       CALL DISPLAY
        GO TO 4

C NOTE:  IN A DEDICATED SYSTEM, MORE ELABORATE STOP INSTRUCTIONS
C        WOULD BE REQUIRED TO SAVE ALL OF THE DATA NOT YET IN THE
C        PERMANANT HISTORY FILE.

C     PRINT FILES FOR COMPARISON WITH PLOTS
4       PRINT 60
60      FORMAT("1","INPUT BUFFER IXY/CURRENT DATA XY/PLOT FILE XY    /AMP&P
       1HA  AP     /AMP&PHA DEV APD/PLOT FILE APD  /PHA DEV FREQ SP/PLOT FI
       2LE PDFS /")
        PRINT 55,(IXY(I,1),XY(I,1),PFXY(I),AP(I,1),APD(I,1),PFAPD(I),PDFS(
       1I,1),PFPDFS(I),I=1,120)
55      FORMAT(" ",16F8.2)

        PRINT 50
50      FORMAT(" ","INPUT BUFFER IXY/CURRENT DATA XY/PLOT FILE XY    /AMP&P
       1HA  AP     /AMP&PHA DEV APD/PLOT FILE APD  /PHA DEV FREQ SP/PLOT FI
       2LE PDFS /")
        PRINT 65,(APT(I),I=1,16)
65      FORMAT("-","TARGET AMPLITUDE AND PHASE    ",2F8.2)


        STOP
        END
      SUBROUTINE STRTAN
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C   START ANALYSIS AND READ AVAILABLE DATA

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSFL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSFL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C   INITIALIZE PARAMETERS, FLAGS, AND CONTROLS
      IF(FENTPAR.EQ.YES) CALL ENTPARM

C    WAIT FOR DATA OR CONTROL INTERUPTS (SIMULATED AT AFIT)
      CALL WAIT

C    CHECK FOR FULL INPUT BUFFER
      IF(FIXYFUL(1).EQ.YES) GO TO 1
      IF(FIXYFUL(2).EQ.YES) GO TO 2
      RETURN

C    READ IN FULL INPUT BUFFER IXY(LN)
1     LN=1
      GO TO 3
2     LN=2
3     CALL DATRDY
      RETURN
      END
```

BEST AVAILABLE COPY

```
      SUBROUTINE ENTPARM
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C ENTER PARAMETERS AND INITIALIZE PROGRAM FLAGS AND CONTROLS

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PEXYT(16),PEXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDES(120,2),PEAPD(120)
      COMPLEX PEPDES(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FOT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PEXYT,PEXY
      COMMON  APT,AP,APD,PDES,PEAPD
      COMMON  PEPDES
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FOT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      YES=1
      NO=0

C  SET COMPRESSION INTERVAL IN SECONDS
      DT=3000

C   SET LINE LNTIME TO START
      LNTIME(1)=0.0
      LNTIME(2)=0.0

C  SET FLAGS TO START
      FCONINT=YES
      FDATINT(1)=YES
      FDATINT(2)=NO
      FLO(1)=NO
      FLO(2)=NO
      FLI(1)=NO
      FLI(2)=NO
      FOT(1)=NO
      FOT(2)=NO
      FDATRDY=NO
      FPLSEL=NO
      FINISH=NO
      FIXYFUL(1)=NO
      FIXYFUL(2)=NO
      FPLXY=NO
      FPLAPD=NO
      FPLPDES=NO
      FPLEMSM=NO
      FPLLNSM=NO
      FPLHRSM=NO
      FPLDASM=NO
```

```
C     ENTER TARGET XY COORDINATES INTO XYT ARRAY
        XYT(1)=CMPLX(0.,+95.)
        XYT(2)=CMPLX(0.,+43.)
        XYT(3)=CMPLX(0.,-43.)
        XYT(4)=CMPLX(0.,-95.)
        XYT(5)=CMPLX(+24.,+24.)
        XYT(6)=CMPLX(+24.,-24.)
        XYT(7)=CMPLX(-24.,-24.)
        XYT(8)=CMPLX(-24.,+24.)
        XYT(9)=CMPLX(+48.,+48.)
        XYT(10)=CMPLX(+48.,-48.)
        XYT(11)=CMPLX(-48.,+48.)
        XYT(12)=CMPLX(-48.,-48.)
        XYT(13)=CMPLX(+95.,0.)
        XYT(14)=CMPLX(+43.,0.)
        XYT(15)=CMPLX(-43.,0.)
        XYT(16)=CMPLX(-95.,0.)

C   CONVERT TARGET XY TO TARGET AP COORDINATES
        DO 1 I=1,16
         A=CABS(XYT(I))
         X=REAL(XYT(I))
         Y=AIMAG(XYT(I))
         P=57.2957795*ATAN2(Y,X)
         APT(I)=CMPLX(A,P)
1       CONTINUE

C NEGATE ENTER PARAMETER FLAG
        FENTPAP=NO

        RETURN
        END
```

BEST AVAILABLE COPY

```
      SUBROUTINE WAIT
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C  WAIT FOR INTERRUPTS (SIMULATED AT AFIT)

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C  DETERMINE DATA OR CONTROL INTERRUPT
1     IF(FDATINT(1).EQ.YES) GO TO 2
      IF(FDATINT(2).EQ.YES) GO TO 3
      IF(FCONINT.EQ.YES) GO TO 5
      GO TO 1

C  READ IN DATA POINT  XY(LN)
2     LN=1
      GO TO 4
3     LN=2
4     CALL DATINT
      IF(FIXYFUL(LN).EQ.YES) GO TO 6
      GO TO 1

C  READ IN CONTROL (DISPLAY SELECTION OR PARAMETER CHANGE)
5     CALL CONINT

6     RETURN
      END
```

BEST AVAILABLE COPY

```
      SUBROUTINE CONINT
C-----------------------------------------------------------------------
C  PROCESS CONTROL INTERRUPT (SIMULATION AT AFIT)

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON   XYT,IXY,XY,PFXYT,PFXY
      COMMON   APT,AP,APD,PDFS,PFAPD
      COMMON   PFPDFS
      COMMON   LN,LNSEL,YES,NO,LNC
      COMMON   FCONINT,FDATINT,FLO,FLI,FDT
      COMMON   FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON   FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON   DT,LNTIME,DAYT,TYME
      COMMON   PLOTMAP(130,130)

C  SET DESIRED DISPLAY FLAGS AND DESIRED LINE
      FPLXY=YES
      FPLAPD=YES
      FPLPDFS=YES
      LNSEL=1

C  SET APPROPRIATE CONTROL FLAGS

      IF(FPLXY+FPLAPD+FPLPDFS+FPLFMSM+FPLLNSM+FPLHRSM+FPLDASM.GE.YES) GO
     1TO 1
      FPLSEL=NO
      GO TO 2
1     FPLSEL=YES
2     FCONINT=NO
      FINISH=YES
      RETURN
      END
```

BEST AVAILABLE COPY

```
      SUBROUTINE DATINT
C-----------------------------------------------------------------------
C  INTERRUPT ROUTINE TO READ IN XY DATA POINT
C  (TEMPORARY TEST ROUTINE)

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLT,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C GENERATE SAMPLE RECEIVED POINTS


      REAL AVE,SIGMA,PJAMP,PJPER,TARGET,NPD,AH,PH
      READ*, AVE
      READ*, SIGMA
      READ*, PJPER
      READ*, PJAMP
      READ*, TARGET
      READ*, NPD
      READ*,AH
      READ*,PH


C     GENERATE 120 SAMPLE RECEIVED DATA POINTS
       PI=3.14159
       DO 10 I=1,120

C     GENERATE RANDOM UNIFORM TARGET SELECTION
        IT=1+16*RANF(B)

C     GENERATE RANDOM GAUSSIAN AMPLITUDE DEVIATION
        AD=0
        DO 20 J=1,12
         DUM=RANF(AD)
         AD=AD+DUM
20       CONTINUE
        AD=(AD-6)*SIGMA-AVE

C     GENERATE RANDOM GAUSSIAN PHASE DEVIATION
        PD=0
        DO 30 K=1,12
         DUM=RANF(PD)
        PD=PD+DUMM
30     CONTINUE
        PD=(PD-6)*SIGMA-AVE

      A=TARGET*REAL(APT(IT))+AD+AH
      P=AIMAG(APT(IT))+PJAMP*COS(2*PI*I/PJPER)+PD*NPD+PH

        IX=A*COS(PI*P/180)
        IY=A*SIN(PI*P/180)
        X=IX
        Y=IY
        IF(ABS(X).GE.128.) X=0
        IF(ABS(Y).GE.128.) Y=0
        IXY(I,1)=CMPLX(X,Y)
10     CONTINUE

C    CLEAR DATA INTERRUPT FLAG
        FDATINT(LN)=NO

C    SET INPUT BUFFER FULL FLAG
        FIXYFUL(LN)=YES

      RETURN
      END
```

BEST AVAILABLE COPY

```
      SUBROUTINE DATRDY
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C MOVE INPUT DATA BUFFER IXY TO CURRENT DATA TABLE XY

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FP_PDFS,FPLEMSM,FPLLNSM,FP_HRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FP_PDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C    CYCLE FULL INPUT BUFFER INTO FILE XY
      DO 1 I=1,120
       XY(I,LN)=IXY(I,LN)
1     CONTINUE

C    CLEAR FULL BUFFER FLAG
      FIXYFUL(LN)=NO

C    SET DATRDY FLAG
     FDATRDY=YES

      RETURN
      END
      SUBROUTINE CURRDAT
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C PROCESS CURRENT DATA

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  FLOTMAP(130,130)

C   CLEAR DATA READY FLAG: SET LNC (LINE # BEING PROCESSED)
      FDATRDY=NO
      LNC=LN

C   CONVERT CARTESIAN TO POLAR COORDINATES
      CALL XYTOPOL

C   CALCULATE AMPLITUDE AND PHASE DEVIATION
      CALL APDEV

C   CALCULATE PHASE DEVIATION FREQUENCY SPECTRUM
      CALL FRQSPEC

      RETURN
      END
      SUBROUTINE XYTOPOL
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C CONVERT CARTESIAN TO POLAR COORDINATES

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C    CONVERSION LOOP
      DO 1 I=1,120
       A=CABS(XY(I,LNC))

C    DETERMINE LINEOUT/LINEIN CONDITION
       IF(A.EQ.0.) GO TO 10
       IF(A.LE.12.0) GO TO 11
       FLI(LNC)=YES
       GO TO 12

C    DEFINE A=0 VALUES AND SET FLO(LNC)
10     A=0. $ P=0.
       FLO(LNC)=YES
       GO TO 13
11     FLO(LNC)=YES
       GO TO 12

C    DETERMINE PHASE
12     X=REAL(XY(I,LNC))
       Y=AIMAG(XY(I,LNC))
       P=57.2957795*ATAN2(Y,X)

C    CREATE AP TABLE
13     AP(I,LNC)=CMPLX(A,P)
1     CONTINUE

      RETURN
      END
      SUBROUTINE APDEV
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C DETERMINE AMPLITUDE AND PHASE DEVIATION

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHPSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      REAL D(16),DMIN

C   CYCLE THROUGH ENTIRE XY TABLE
       DO 1 I=1,120

C    CHECK FOR LO CONDITION
       IF(REAL(AP(I,LNC)).LE.12.) GO TO 3

C    DETERMINE DISTANCE FROM EVERY TARGET
       DO 2 J=1,16
        D(J)=CABS(XY(I,LNC)-XYT(J))
2      CONTINUE

C    DETERMINE MINIMUM DISTANCE
       DMIN=AMIN1(D(1),D(2),D(3),D(4),D(5),D(6),D(7),D(8),D(9),D(10),D(1
     11),D(12),D(13),D(14),D(15),D(16))

C    DETERMINE NEAREST TARGET
       DO   J=1,16
        IF(DMIN.NE.D(J)) GO TO 4

C    CALCULATE AMPLITUDE AND PHASE DEVIATION
        APD(I,LNC)=AP(I,LNC)-APT(J)

C    RESOLVE + OR- PI AMBIGUITY
       IF(AIMAG(APD(I,LNC)).LE.-180.) APD(I,LNC)=CMPLX(REAL(APD(I,LNC)),
     1AIMAG(APD(I,LNC))+360.)
4      CONTINUE
       GO TO 1

C    ASSIGN APD VALUES FOR LO CONDITION
3      APD(I,LNC)=CMPLX(30.,30.)

1      CONTINUE
      RETURN
       END
      SUBROUTINE FRQSPEC
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C DETERMINE FREQUENCY SPECTRUM OF PHASE DEVIATION (FFT)

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      COMPLEX PD(128),U,W,T

      PI=3.14159265

C    CYCLE DATA INTO PD ARRAY TO BE TRANSFORMED
       DO 1 I=1,120
        PD(I)=CMPLX(AIMAG(APD(I,LNC)),0.)
1      CONTINUE

C    ZERO REMAINDER OF PD ARRAY
       DO 2 I=121,128
        PD(I)=CMPLX(0.,0.)
2      CONTINUE

C    PERFORM APPROPRIATE BIT REVERSAL OF PD ARRAY
       J=1
       DO 3 I=1,127
        IF(I.GE.J) GO TO 10
        T=PD(J)
        PD(J)=PD(I)
        PD(I)=T

10      K=64
11      IF(K.GE.J) GO TO 12
        J=J-K
        K=K/2
        GO TO 11

12      J=J+K
3      CONTINUE

C    PERFORM TRANSFORMATION
       DO 4 L=1,7
        LE=2**L
```

BEST AVAILABLE COPY

```
          LE1=LE/2
          U=CMPLX(1.,0.)
          W=CMPLX(COS(PI/FLOAT(LE1)),-SIN(PI/FLOAT(LE1)))

          DO 5  J=1,LE1
           DO 6  I=J,128,LE
           IP=I+LE1
           T=PD(IP)*U
           PD(IP)=PD(I)-T
           PD(I)=PD(I)+T
6          CONTINUE
          U=U*W
5         CONTINUE
4        CONTINUE

C     PUT TRANSFORMED DATA BACK INTO CURRENT DATA FILE
         DO 7  I=1,120
          PDES(I,LNC)=PD(I)
7        CONTINUE

         RETURN
         END
```

```
      SUBROUTINE RECRDAT
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C RECORD DATA INTO PERMANENT FILE IN TIME COMPRESSED FORM

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C   COMPRESS DATA
      CALL COMPRES

C   DETERMINE LINE READY FOR PERMANENT FILE
      IF(FDT(1).EQ.YES) GO TO 1
      IF(FDT(2).EQ.YES) GO TO 1
      RETURN

C   PUT COMPRESSED DATA INTO PERMANENT FILE

1     CALL HISTORY
      RETURN
      END
      SUBROUTINE COMPRES
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C COMPRESS DATA

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATTINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
```

```
      SUBROUTINE HISTORY
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C HISTORY

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC(2)
      INTEGER FCONINT,FDATTINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
      SUBROUTINE DISPLAY
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C AFFERENT CONTROL ROUTINE

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      CALL SELDAT
      CALL SELDIS
      RETURN
      END
      SUBROUTINE SELDAT
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C SELECTION OF SELECT DATA ROUTINE

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      IF(FPLXY+FPLAPD+FPLPDFS.GE.YES) GO TO 500
      IF(FPLEMSM.EQ.YES) GO TO 501
      IF(FPLLNSM.EQ.YES) GO TO 502
      IF(FPLHRSM.EQ.YES) GO TO 503
      IF(FPLDASM.EQ.YES) GO TO 504
      RETURN
500   CALL SLCURR
      RETURN
501   CALL SLEMSM
      RETURN
502   CALL SLLNSM
      RETURN
503   CALL SLHRSM
      RETURN
504   CALL SLDASM
      RETURN
      END
```

BEST AVAILABLE COPY

```
      SUBROUTINE SLCURR
C--------------------------------------------------------------------------
C SELECT CURRENT DATA FOR DISPLAY

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FOT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FOT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

C  DETERMINE DATE AND TIME
      CALL DATE(DAYT)
      CALL TIME(TYME)

C   SELECT APPROPRIATE CURRENT DATA AND TRANSFER TO PLOT FILES
      IF(FPLXY.EQ.NO) GO TO 12
      DO 10 I=1,16
       PFXYT(I)=XYT(I)
10    CONTINUE
      DO 11 I=1,120
       PFXY(I)=XY(I,LNSEL)
11    CONTINUE
12    IF(FPLAPD.EQ.NO) GO TO 14
      DO 13 I=1,120
       PFAPD(I)=APD(I,LNSEL)
13    CONTINUE
14    IF(FPLPDFS.EQ.NO) GO TO 16
      DO 15 I=1,120
       PFPDFS(I)=PDFS(I,LNSEL)
15    CONTINUE
16    RETURN
      END
```

```
C-----------------------------------------------------------------------
C DUMMY SELECT DATA SUBROUTINES NOT YET DEVELOPED

      SUBROUTINE SLEMSM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PEXYT(16),PEXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDES(120,2),PEAPD(120)
      COMPLEX PEPDES(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PEXYT,PEXY
      COMMON  APT,AP,APD,PDES,PEAPD
      COMMON  PEPDES
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END






      SUBROUTINE SLLNSM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PEXYT(16),PEXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDES(120,2),PEAPD(120)
      COMPLEX PEPDES(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PEXYT,PEXY
      COMMON  APT,AP,APD,PDES,PEAPD
      COMMON  PEPDES
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDES,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
      SUBROUTINE SLHRSM
C-----------------------------------------------------------------------
      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FP_PDFS,FPLFMSM,FPLLNSM,FP_HRSM,FPLOASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON   XYT,IXY,XY,PFXYT,PFXY
      COMMON   APT,AP,APD,PDFS,PFAPD
      COMMON   PFPDFS
      COMMON   LN,LNSEL,YES,NO,LNC
      COMMON   FCONINT,FDATINT,FLO,FLI,FDT
      COMMON   FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON   FPLXY,FP_PDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLOASM,FPLAPD
      COMMON   DT,LNTIME,DAYT,TYME
      COMMON   PLOTMAP(130,130)

      RETURN
      END
```

```
      SUBROUTINE SLOASM
C-----------------------------------------------------------------------
      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FP_SEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FP_LNSM,FPLHRSM,FPLOASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON   XYT,IXY,XY,PFXYT,PFXY
      COMMON   APT,AP,APD,PDFS,PFAPD
      COMMON   PFPDFS
      COMMON   LN,LNSEL,YES,NO,LNC
      COMMON   FCONINT,FDATINT,FLO,FLI,FDT
      COMMON   FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON   FPLXY,FP_PDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLOASM,FPLAPD
      COMMON   DT,LNTIME,DAYT,TYME
      COMMON   PLOTMAP(130,130)

      RETURN
      END
      SUBROUTINE SELDIS
C------------------------------------------------------------------------
C------------------------------------------------------------------------
C SELECTION OF DISPLAY OUTPUT ROUTINE

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      IF(FPLXY+FPLAPD+FPLPDFS.GE.YES) GO TO 510
      IF(FPLEMSM.EQ.YES) GO TO 511
      IF(FPLLNSM.EQ.YES) GO TO 512
      IF(FPLHRSM.EQ.YES) GO TO 513
      IF(FPLDASM.EQ.YES) GO TO 514
      RETURN
510   IF(FPLXY.EQ.YES) CALL DISXY
      IF(FPLAPD.EQ.YES) CALL DISAPD
      IF(FPLPDFS.EQ.YES) CALL DISPDFS
      RETURN
511   CALL DISEMSM
      RETURN
512   CALL DISLNSM
      RETURN
513   CALL DISHRSM
      RETURN
514   CALL DISDASM
      RETURN
      END
```

"BEST AVAILABLE COPY

```
      SUBROUTINE DISXY
C---------------------------------------------------------------------
C DISPLAY OF SIGNAL SPACE DATA
      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLOASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLT,FDT
      COMMON  FDATRDY,FPLSEL,FTNISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLOASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      REAL MAPXY(129,129)
      COMMON MAPXY
      DATA TARGET/1HT/,STAR/1H*/,HOR/1H-/,VER/1HI/,BLANK/1H /

C   SET MAPXY BLANK
      DO 11 I=1,129
       DO 10 J=1,129
         MAPXY(I,J)=BLANK
10     CONTINUE
11    CONTINUE

C   SET MAPXY HOR GRID
      DO 13 I=1,129
       DO 12 J=1,129,32
         MAPXY(I,J)=HOR
12     CONTINUE
13    CONTINUE

C   SET MAPXY VER GRID
      DO 15 I=1,129,32
       DO 14 J=1,129
         MAPXY(I,J)=VER
14     CONTINUE
15    CONTINUE

C   SET MAPXY VER SCALE
      DO 17 I=1,129,32
       DO 16 J=1,129,8
         MAPXY(I,J)=HOR
16     CONTINUE
17    CONTINUE

C   SET MAPXY HOR SCALE
      DO 19 I=1,129,8
       DO 18 J=1,129,32
         MAPXY(I,J)=VER
```

BEST AVAILABLE COPY

```
18       CONTINUE
19      CONTINUE

C   ASSIGN DATA STARS TO MAPXY
        DO 20 I=1,120
         PFXY(I)=(PFXY(I))/2
         IX=REAL(PFXY(I))
         IY=AIMAG(PFXY(I))
         MAPXY(65+IX,65-IY)=STAR
20      CONTINUE

C   ASSIGN TARGETS TO MAPXY
        DO 21 I=1,16
         PFXYT(I)=(PFXYT(I))/2
         IX=REAL(PFXYT(I))
         IY=AIMAG(PFXYT(I))
          MAPXY(65+IX,65-IY)=TARGET
21       CONTINUE

C   EJECT TO TOP OF NEXT PAGE AND PRINT TITLE:  SHIFT TO 8 VER LN/INCH
        PRINT 50,LNSEL,DAYT,TYME
50      FORMAT("1","CURRENT DATA SIGNAL SPACE: LINE:",I2," ; DATE:",A10,";
      1 TIME:",A10)
        PRINT 55
55      FORMAT("T")
        PRINT 70
70      FORMAT(2X//2X)

C   PRINT GRAPH DISXY TOP SCALE
        PRINT60
60      FORMAT(2X,"    -128     -112      -96      -80      -64      -48      -32
      1     -16       0      +16      +32      +48      +64      +80      +96
      2  +112     +128")

C   PRINT GRAPH
        MARK=7
        DO 100 MLN=1,129
         IF(MARK.EQ.7) GO TO 58
         IF(MLN.EQ.45) GO TO 56
         IF(MLN.EQ.46) GO TO 57
         GO TO 59

C       PRINT WITH VER SCALE NUMBERS
58        MVS=130-2*MLN
          PRINT 61,MVS,(MAPXY(I,MLN),I=1,129)
61        FORMAT(3X,I4,129A1)
          MARK=0
          GO TO 100

C       PRINT WITH VER AXIS LABEL "Y"
56        PRINT 62,(MAPXY(I,MLN),I=1,129)
62        FORMAT(3X,1HY,3X,129A1)
           MARK=MARK+1
          GO TO 100

C       PRINT WITH VER AXIS LABEL "AXIS"
57        PRINT 63,(MAPXY(I,MLN),I=1,129)
```

```
63      FORMAT(2X,4HAXIS,1X,129A1)
        MARK=MARK+1
        GO TO 100

C     PRINT PLAIN MAPXY
59      PRINT 64,(MAPXY(I,MLN),I=1,129)
64      FORMAT(7X,129A1)
        MARK=MARK+1
        GO TO 100

100    CONTINUE

C   PRINT BOTTOM SCALE AND AXIS LABEL
       PRINT 60
       PRINT 65
65     FORMAT("S")
       PRINT 66
66     FORMAT(" ",10X,"X AXIS")

C CLEAR DISXY FLAG "FPLXY"
       FPLXY=NO
       RETURN
       END
```

BEST AVAILABLE COPY

```
      SUBROUTINE DISAPD
C____________________________________________________________________________
C DISPLAY OF AMPLITUDE AND PHASE DEVIATION

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FP_PDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FP_PDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME

      REAL MAPAPD(122,121)
      COMMON MAPAPD
      DATA AMP/1HA/,PHA/1HP/,HOR/1H-/,VER/1H!/,BLANK/1H /,LO/2HLO/
      DATA ZL/1HL/,ZI/1HI/,ZN/1HN/,E/1HE/,O/1HO/,U/1HU/,T/1HT/

C SET MAPAPD BLANK
      DO 1 I=1,122
       DO 2 J=1,121
        MAPAPD(I,J)=BLANK
2      CONTINUE
1     CONTINUE

      DO 3 I=1,122
      DO 4 J=1,121,12
C SET MAPAPD HOR GRID
        MAPAPD(I,J)=HOR
4      CONTINUE
3     CONTINUE

C SET MAPAPD VER GRID
      DO 5 J=1,121
       DO 6 I=1,61,10
        MAPAPD(I,J)=VER
6      CONTINUE
       DO 7 I=62,122,10
        MAPAPD(I,J)=VER
7      CONTINUE
5     CONTINUE

C ASSIGN APD DATA TO MAPAPD
      DO 8 I=1,120
       IF(REAL(PFAPD(I)).GE.28.) GO TO 10
       A=REAL(PFAPD(I))
        P=AIMAG(PFAPD(I))
       IA=
      IP=P
      MAPAPD(IA+31,I)=AMP
      MAPAPD(IP+92,I)=PHA
      GO TO 8

10    MAPAPD(58,I)=ZL
      MAPAPD(59,I)=ZI
      MAPAPD(60,I)=ZN
      MAPAPD(61,I)=E
      MAPAPD(62,I)=BLANK
      MAPAPD(63,I)=O
      MAPAPD(64,I)=U
      MAPAPD(65,I)=T

8     CONTINUE

C EJECT TO TOP OF NEXT PAGE AND PRINT TITLE; SHIFT TO 8 VER LN/INCH
      PRINT 50,LNSEL,DAYT,TYME
50    FORMAT("1","CURRENT DATA AMPLITUDE AND PHASE DEVIATIONS; LINE:",I2
     1," : DATE:",A10,": TIME:",A10)
      PRINT 55
55    FORMAT("T")

C PRINT GRAPH LEGEND
      PRINT 60
60    FORMAT("0","  A=AMPLITUDE DEVIATION:    P=PHASE DEVIATION")

C PRINT TOP SCALING INFORMATION
      PRINT 65
65    FORMAT("0",10X,"SCALED WITH MODEM X,Y VALUES")
      PRINT 70
70    FORMAT(" ",9X,"-30        -20        -10         0       +10        +
     120       +30!!-30       -20        -10          0 DEG.   +10        +20
     2    +30")

C PRINT GRAPH DO LOOP
      MARK=11
      IYV=-5
      DO 9 MLN=1,121
       IF(MARK.EQ.11) GO TO 101
       IF(MLN.EQ.50) GO TO 102
       IF(MLN.EQ.51) GO TO 103

C   PRINT PLAIN GRAPH
      PRINT 75,(MAPAPD(I,MLN),I=1,122)
75    FORMAT(11X,122A1)
      MARK=MARK+1
      GO TO 9

C   PRINT GRAPH WITH SIDE SCALE
101   IYV=IYV+5
      PRINT 80,IYV,(MAPAPD(I,MLN),I=1,122)
80    FORMAT(9X,I2,122A1)
      MARK=0
      GO TO 9

C   PRINT GRAPH WITH SIDE AXIS LABEL
```

BEST AVAILABLE COPY

```
102     PRINT 85,(MAPAPD(I,MLN),I=1,122)
85      FORMAT(1X,"TIME IN",3X,122A1)
        MARK=MARK+1
        GO TO 9
103     PRINT 90,(MAPAPD(I,MLN),I=1,122)
90      FORMAT(1X,"MILLISEC.",1X,122A1)
        MARK=MARK+1
        GO TO 9

C    END PRINT LOOP
9        CONTINUE

C REPRINT BOTTOM SCALING INFORMATION

        PRINT 70
        PRINT 65

C REPRINT LEGEND AT BOTTOM

        PRINT 60

C RETURN TO STANDARD LINE SPACING

        PRINT 95
95      FORMAT("S")

C CLEAR DISPLAY APD FLAG
        FPLAPD=NO

        RETURN
        END
```

```
      SUBROUTINE DISPDFS
C-------------------------------------------------------------------------
C DISPLAY OF PHASE DEVIATION FREQUENCY SPECTRUM

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      DATA STAR/1H*/,VER/1H!/,HOR/1H-/,BLANK/1H /

C     SET PLOTMAP BLANK
        DO 1 I=1,121
         DO 2 J=1,121
          PLOTMAP(I,J)=BLANK
2        CONTINUE
1       CONTINUE

C     SET HOR PLOTMAP GRID
        DO 3 I=1,121
         DO 4 J=1,121,12
          PLOTMAP(I,J)=HOR
4        CONTINUE
3       CONTINUE

C     SET VER PLOTMAP GRID
        DO 5 I=1,121,20
         DO 6 J=1,121
          PLOTMAP(I,J)=VER
6        CONTINUE
5       CONTINUE

C     SET PDFS PLOT DATA INTO PLOTMAP
        DO 7 J=1,64
         FMAG=CABS(PFPDFS(J))
         MAG=FMAG/(500./120.)
         IF(MAG.GE.120) MAG=119
         K=0.9375*J
         PLOTMAP(K+2,121-MAG)=STAR
7       CONTINUE

C     EJECT TO TOP OF NEXT PAGE AND PRINT TITLE: SHIFT TO 8 VER LN/INCH
        PRINT 60,LNSEL,DAYT,TYME
```

```
50     FORMAT("1","CURRENT DATA PHASE DEVIATION FREQUENCY SPECTRUM; LINE
      1",I2," ; DATE:",A10,": TIME:",A10)
       PRINT 55
55     FORMAT("T")

C     PRINT TOP LABEL
       PRINT 60
60     FORMAT("0",15X,"FREQUENCY IN HERTZ")

C     PRINT TOP SCALE
       PRINT 65
65     FORMAT("0",15X,"0",16X,"200",17X,"400",17X,"600",17X,"800",16X,"10
      100",15X,"1200")

C     PRINT GRAPH LOOP, SIDE SCALE, AND LABEL
       MARK=11
       IYV=550
       DO 8 MLN=1,121
        IF(MARK.EQ.11) GO TO 21
        IF(MLN.EQ.54) GO TO 24
        IF(MLN.EQ.53) GO TO 23
        IF(MLN.EQ.52) GO TO 22

C      PRINT PLAIN GRAPH
        PRINT 75,(PLOTMAP(I,MLN),I=1,121)
75      FORMAT(16X,121A1)
        MARK=MARK+1
        GO TO 8

C      PRINT GRAPH WITH SIDE LABEL
22     PRINT 80,(PLOTMAP(I,MLN),I=1,121)
80     FORMAT(1X,"FREQUENCY",6X,121A1)
        MARK=MARK+1
        GO TO 8


23      PRINT 85,(PLOTMAP(I,MLN),I=1,121)
85      FORMAT(1X,"SPECTRUM",7X,121A1)
        MARK=MARK+1
        GO TO 8

24      PRINT 90,(PLOTMAP(I,MLN),I=1,121)
90      FORMAT(1X,"MAGNITUDE",6X,121A1)
        MARK=MARK+1
        GO TO 8

C      PRINT GRAPH WITH SIDE AXIS SCALE
21      IYV=IYV-50
        PRINT 95,IYV,(PLOTMAP(I,MLN),I=1,121)
95      FORMAT(1X,11X,I4,121A1)
        MARK=0
        GO TO 8

C      END GRAPH LOOP
8       CONTINUE

C     REPRINT BOTTOM SCALE
```

BEST AVAILABLE COPY

```
      PRINT 65
C     REPRINT BOTTOM LABEL
      PRINT 60
C     RETURN TO STANDARD LINE SPACING
      PRINT 100
100   FORMAT("S")
C     CLEAR DISPLAY FLAG
      FELPDFS=NO
      RETURN
      END
```

```
C-----------------------------------------------------------------------
C DUMMY DISPLAY SUBROUTINES NOT YET DEVELOPED

      SUBROUTINE DISFMSM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END




      SUBROUTINE DISHRSM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLFMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
      SUBROUTINE DISLNSM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
```

```
      SUBROUTINE DISDASM
C-----------------------------------------------------------------------

      COMPLEX XYT(16),IXY(120,2),XY(120,2),PFXYT(16),PFXY(120)
      COMPLEX APT(16),AP(120,2),APD(120,2),PDFS(120,2),PFAPD(120)
      COMPLEX PFPDFS(120)
      INTEGER LN,LNSEL,YES,NO,LNC
      INTEGER FCONINT,FDATINT(2),FLO(2),FLI(2),FDT(2)
      INTEGER FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL(2)
      INTEGER FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      REAL DT,LNTIME(2)
      COMMON  XYT,IXY,XY,PFXYT,PFXY
      COMMON  APT,AP,APD,PDFS,PFAPD
      COMMON  PFPDFS
      COMMON  LN,LNSEL,YES,NO,LNC
      COMMON  FCONINT,FDATINT,FLO,FLI,FDT
      COMMON  FDATRDY,FPLSEL,FINISH,FENTPAR,FIXYFUL
      COMMON  FPLXY,FPLPDFS,FPLEMSM,FPLLNSM,FPLHRSM,FPLDASM,FPLAPD
      COMMON  DT,LNTIME,DAYT,TYME
      COMMON  PLOTMAP(130,130)

      RETURN
      END
```

```
  CSB     NOS/BE L414I         CYBR CMR3 08/31/77
14.36.53.MIN3832  FROM    /38
14.36.53.IP  00005568 WORDS - FILE INPUT  , DC 00
14.36.53.MIN,CM100000,STCSB. T770269,MINTONYE,GE7
14.36.53.70,BOX 4052
14.36.55.FTN,B=PUNCHB,OPT= ,R=0.
14.39.07.  NULL PROGRAM IGNORED AFTER  CISDASM
14.39.07.      6.247 CP SECONDS COMPILATION TIME
14.39.07.MAP,PART.
14.39.07.LOAD,PUNCHB.
14.39.07.EXECUTE.
14.39.07.     STOP
14.39.07.        .874 CP SECONDS EXECUTION TIME
14.39.07.OP  00003648 WORDS - FILE PUNCHB , DC 10
14.39.07.OP  00017152 WORDS - FILE OUTPUT , DC 40
14.39.07.MS  136192  WORDS  (  136192 MAX USED)
14.39.07.SCM  71630  WORDS MAXIMUM
14.39.07.CPA       8.344 SEC.           3.622 ADJ.
14.39.07.IO       19.958 SEC.           9.984 ADJ.
14.39.07.CM      646.515 KWS.           5.173 ADJ.
14.39.07.CRUS                          18.750
14.39.07.COST                     $     1.12
14.39.07.PP       49.440 SEC.      DATE 10/15/77
14.39.07.EJ  END OF JOB, 38  T770269.

*****.*****          MIN3832   //// END  OF LIST ////
```

BEST AVAILABLE COPY

32

VITA

Robert A. Mintonye was born on 25 July 1946, in Coos Bay, Oregon. He graduated from high school at Coquille, Oregon, in 1964, and attended Oregon State University where he obtained a Bachelor of Science degree for Mechanical Engineering in June 1969. Also at that time, he was commissioned into the United States Air Force through the ROTC program. In October 1969, he was sent to the Officers Communications School at Keesler Air Force Base, Mississippi. Upon his graduation from the nine month school as a Communications Maintenance Officer (AFSC 3034), he was assigned to the 1828 Electronics Installation Squadron at Wright Patterson Air Force Base, Ohio, where he performed team chief duties on radio and construction projects. In 1971, he was assigned to the European Communications Area Headquarters as a program manager of engineering and installations for wide band communications systems. He worked on various microwave and tropospheric scatter radio projects in Germany until May 1975, when he was selected for AFIT to begin the Electrical Engineering Masters of Science program.

G

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/77-28	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Quantitative Processing of Signal Space Data to Provide Transmission Line Analysis		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Robert A. Mintonye Capt. USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Communications and Control Division (RADC-DCLF) Rome Air Development Center Griffiss AFB, New York 13441		12. REPORT DATE December 1977 13. NUMBER OF PAGES 131
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17

[illegible]

JERRAL F. GUESS, CAPTAIN, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Digital Signal Space Analysis
Transmission Line Analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Quantitative processing of signal space data to provide transmission line analysis was developed in support of work being done at the USAF Rome Air Development Center Laboratories. The RADC engineers were working with digital modems and they were qualitatively analyzing transmission line perturbations through observation of an oscilloscope display of the signal space representation. The display was being generated by an optional circuit available with the modem, but the oscilloscope display was not capable of providing adequate quantitative perturbation information. However, the signal space data did contain all of the information required for a complete quantivative analysis of the transmission line perturbations. Therefore, a system using the same data potentially could be devised to perform the analysis. A signal space data analysis system design was developed in this thesis, which would be capable of providing three displays of the signal space representation, the amplitude and phase deviations, and the frequency spectrum of the phase deviations or phase jitter. The system would consist of the digital modem, a minicomputer, and an interface device between the modem and minicomputer which would be capable of interrupting the minicomputer. A FORTRAN program was also developed and run in a simulation effort which transformed the xy data of the signal space into a form of data which could be plotted on the three displays. Though the system was not actually constructed, it was successfully simulated such that it would merit implementation.